Sorbonne Université — LIP6 & ISIR — EDITE de Paris

THÈSE

défendue par

Étienne Simon

en vue de l'obtention du grade de Docteur

Deep Learning for Unsupervised Relation Extraction

---

soutenue publiquement le 5 juillet 2022

Devant le jury composé de

Pr Alexandre Allauzen — Rapporteur
Professeur des universités, Université Paris-Dauphine PSL, ESPCI
Dr Benoit Favre — Rapporteur
Maître de conférences, Aix-Marseille Université
Pr Pascale Sébillot — Examinatrice
Professeure des universités, IRISA, INSA Rennes
Pr Xavier Tannier — Président
Professeur des universités, Sorbonne Université
Dr Benjamin Piwowarski — Directeur
Chargé de recherche, CNRS, Sorbonne Université
Dr Vincent Guigue — Directeur
Maître de conférences, Sorbonne Université

Abstract

Capturing concepts' interrelations is a fundamental of natural language understanding. It constitutes a bridge between two historically separate approaches of artificial intelligence: the use of symbolic and distributed representations. However, tackling this problem without human supervision poses several issues, and unsupervised models have difficulties echoing the expressive breakthroughs of supervised ones. This thesis addresses two supervision gaps we identified: the problem of regularization of sentence-level discriminative models and the problem of leveraging relational information from dataset-level structures.

The first gap arises following the increased use of discriminative approaches, such as deep neural network classifiers, in the supervised setting. These models tend to collapse without supervision. To overcome this limitation, we introduce two relation distribution losses to constrain the relation classifier into a trainable state. The second gap arises from the development of dataset-level (aggregate) approaches. We show that unsupervised models can leverage a large amount of additional information from the structure of the dataset, even more so than supervised models. We close this gap by adapting existing unsupervised methods to capture topological information using graph convolutional networks. Furthermore, we show that we can exploit the mutual information between topological (dataset-level) and linguistic (sentence-level) information to design a new training paradigm for unsupervised relation extraction.

Acknowledgements

I'm not sure how to write these acknowledgements who to thank how to thank them there are so many people who contributed to the completion of this thesis maybe I should thank my optician who enabled me to read so many papers but why stop there I'm sure my glasses were made using some kind of polishing machine with bolts made by a worker supported through their childhood by a sweetheart now long forgotten I'm however deeply thankful to the boltmaker's childhood sweetheart for bringing them a bit of warmth that might not have been strictly necessary to the conception of the metal fasteners indirectly ensuring my optical prowesses but that I honor anyway it might be easier to list people I don't want to thank no that would be ill-disposed I won't go into woollen coats I'll focus on people more closely related to my doctoral endeavors first of which are my supervisors thank you Benjamin Piwowarski for providing key insight on the information extraction field the gentle encouragements and for the guidance through the years thank you Vincent Guigue for your vast knowledge of how things are done and how they are not I hope you do not disapprove too much of the otters in my defense's slides I'm also grateful to other members of the jury for examining my thesis in particular I would like to thank the reviewers for thoroughly reading my dissertation I hope it was as pleasant to read as it was to write I should elucidate that this is not a curse I quite enjoyed taking the time to put in writing what I learned over the last few years I shall also thank Carl Marletti creator of the Lily Valley with its indecently delicious or maybe deliciously indecent association of sugar fat and violets into a mean of reaching my goal of contracting diabetes on my deathbed note however that since it was contracted through the ingestion of violets it can surely be analyzed as poetical diabetes although still classified as type 2 Carl Your glory in what was an undeniable and mouthwatering emotional support shall not be forgotten I was furthermore corrupted into the sin of gluttony by Sadaharu Aoki Toque Cuivrée and Merveilleux de chez Fred I also enjoy Baillardran despite its overpricing and overtly bourgeois styling I should therefore not be too vocal about it in case liberation precedes alopecia I feel like I'm digressing about boltmakers again no if I have to thank someone for emotional support first and foremost that would be Raphaëlle Labarrière Syrielle Montariol and Billur Sezgin thank you for the numerous evenings and nights spent rebuilding the world and ourselves with an excessive amount of wine thank you for making me discover things that I enjoy about life in particular thank you for sharing a copious amount of literary works I particularly enjoy reading classical authors whose remains are in various states of decomposition between the retirement home and the Panthéon I had the pleasure of snatching several volumes from their skeletal hands thank you Raphaëlle for providing flawless directions in grave robbery I further thank Jill-Jênn Vie and once again Syrielle Montariol and Arij Riabi for proofreading my dissertation you did a great job correcting me and oh boy did I need it thankfully the first version was not witnessed by the official reviewers that's what's called taking one for the team next in order I would like to thank tomatoes in all their shapes and forms thank you for helping me keep a semblance of health for giving a bit more of a culinary personality beyond that of a sugar-addict even though I still have a kind of obsessive relationship with you I'm not ready to deconstruct it yet I wrote you a poem it goes like this Oh tomato Ô tomato 御tomato speaking of health thank you Dr Bouteille for taking care of me and having a funny name and now speaking of bottle I would like to thank sweet-tooth Syrielle Montariol and mixologist Kenza Jernite for being among the most admirable people I know the world is more kind with you in it I enjoyed sharing liquorous wine and cocktails with you and I have faith we will be getting diabetes together with moderation because on the other side Billur is accompanying me beyond moderation on cathartic disdain I can only hope we'll be getting to hell together but for now waiting for this unavoidable and deserved end I would like to thank my bed for being the nicest place on earth and not only my bed all the beds that sheltered me and those that will accommodate me in the future I shall even include couches or any surfaces appropriate for napping I long the days we were forced to sleep in kindergarten and I can only hope this will be generalized to latter stages of life as I'm sleep-deprivedly writing these acknowledgements the night before my defense you might think this clinophilia is revealing of an underlying pathology but fear not as I enjoy all soft things made from fabric I take the opportunity to thank the kind overseeing of my work by communist BERT and the transitional support of Waza Brick Oishii et al despite your stoic airs I know how you feel deep inside I should also thank several members of the Felidae family for providing purring especially Ozwin you're a coward but you're cute you're a cute coward you're worth your weight of catnip when you get the chance of meeting her in a courageous disposition you can hold her little furry paws in your hand feel the softness of this unique specimen yes at times though her dark side overwhelms her and wounds you while you're in the most playful mood she nonchalantly write a Greek tragedy in the thunderous strength of her claws yes yes but cutely though speaking of family I should thank mine which boringly is Hominidae the end of the genealogy tree looks like yes at depth 2 it's still a tree that was an intelligent decision by my ancestors and though I don't remember everything I am thankful for your support early in life I would also like to thank the action de groupe and its 群作用 extension in particular thank you Jill-Jênn for jill-jênning all along thank you Shloub for laughing at my poorly crafted puns or at anything really you're too good of a public thank you Tito for indirectly teaching me more about machine learning than what you yourself know thank you Alex for teaching me more about machine learning than what you wish you knew thank you Link Mauve for showing me you can sluggishly not care about unimportant things thank you Ryan for the gentle squabble and thank you again Tito Alex and Link Mauve for your substantiated subversion speaking of subversion I would like to thank the whole open source community and more broadly organizations encouraging the sharing of information with as many people as possible such as Wikipedia and Sci-Hub but I don't need to go this far for finding people sharing ideas I would like to thank all members of the MLIA team for teaching me in particular a huge thank to the people who participated in our reading groups that was legitimately the best work-related moments during my time in the lab if we go into non-work-related we might end up in some incongruous raclette–karaoke night I would also like to thank the people in the bestest office ever 26–00/534 thank you Marie for your strong laughter Agnès for your strong chill Tristan for your strong flow and Jean-Yves for your strong temporal consistency I'm sure the time at which you leave for your afternoon collation could have been used to calibrate atomic clocks thus providing an unyielding beacon of stability in research's messy life finally thank you Christophe Bouder for dealing with deep learning ludicrous computational requirements and for providing a serious challenge in foosball this remind me that I should also thank my sport mates Syrielle and 26–00/534 for bouldering and Arij for swimming a deep thanks to the municipal employees who decided on the nearly 30 degrees Celsius temperature for the swimming pool I wonder whether you might have been doing more for my physical health than a bottle lost in a municipal health center I take the opportunity to thank the love of my life with whom I share everyday hot water I sing your praise everyday a big thank you to Billur Sezgin for lending me her bathtub and thank you to Lush for achieving the feat of making it more enjoyable thank you also to Manon Dumas Morès for accepting the same bathtub-lending deal I might become a bathtub tycoon in the future I partly grew up in a thermal city maybe that explains why I like hot water so much I'm not sure but just in case thank you to my thermal city Bagnères-de-Luchon to hell with it thank you to all thermal cities they deserve it well maybe one instance of enjoyable cold water was a night bath in Kyoto for Gozan no Okuribi thank you to the protagonist of this otherworldly night for making it so memorable the cold water appeared first in the Takasegawa channel near Pontocho then in the duck's river Kamogawa which despite its name was void of ducks which might have hidden following their awareness of a cooking intent originating in a native of southwestern France thank you to Syrielle Kenza and Manon for participating in singular culinary experiments this is what inspired the OuCuiPo illustration in my introduction sorry Manon though for getting you sick with some weird black pepper ok it's getting late thank you Arthur Suspene for getting old as fast as I do but slightly earlier thank you to Sappho's friend who shall sadly remain unnamed welcome to Jill-Jênn's firework I'm sure it's going to turn out better than the movie and long life to Anne Émone and all her children I want to leave her offspring in 26–00/534 so that we can share something beyond our PhD but I fear most of them will starve to death by the end of the month finally thank you to all the excellent teachers who accompanied me until the end of my formal studies I aspire to be half as good as you were.

Michael: Yes—it wasn't logical.
George : You were a tomato! A tomato doesn't have logic. A tomato can't move.

—"Tootsie" (1982)

This disaster of the Cherokees, brought to me by a sad friend to blacken my days and nights! I can do nothing; why shriek? why strike ineffectual blows? I stir in it for the sad reason that no other mortal will move, and if I do not, why, it is left undone. The amount of it, to be sure, is merely a scream; but sometimes a scream is better than a thesis.

—Ralph Waldo Emerson "Letter to President van Buren" (1838)

Aaaaaaaaaaaah

—Alain Chabat in "Reality" by Quentin Dupieux (2014)

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

ACE	Automatic Content Extraction (Section C.1)
ACL	Association for Computational Linguistics
ARI	Adjusted Rand Index (Section 2.5.1.1)
BERT	Bidirectional Encoder Representations from Transformers (Section 1.3.4)
BPE	Byte-Pair Encoding (Section 1.2.3)
BPR	Bayesian Personalized Ranking (Section 2.4.3)
CNN	Convolutional Neural Network (Section 1.3.1)
DAE	Denoising AutoEncoder (Section 2.5.7)
DARPA	Defense Advanced Research Projects Agency (Section C.4)
DEC	Deep Embedded Clustering (Section 2.5.7)
DIPRE	Dual Iterative Pattern Relation Expansion (Section 2.3.2)
DIRT	Discovery of Inference Rules from Text (Section 2.3.3)
ELBO	Evidence Lower BOund (Section 2.5.5)
ELMo	Embeddings from Language Model (Section 1.3.2.2)
EPGNN	Entity Pair Graph Neural Network (Section 2.4.5)
FB	FreeBase (Section C.3)
GAT	Graph ATtention network (Section 4.3.3)
GCN	Graph Convolutional Network (Section 4.3)
GNN	Graph Neural Network (Section 4.3)
GIS	Generalized Iterative Scaling (Section 2.3.4)
GPE	Geo-Political Entity (Section 2.5.3)
GRU	Gated Recurrent Unit (Section 1.3.2.1)
IDF	Inverse Document Frequency (Section 2.5.3)
JSD	Jensen–Shannon Divergence (Section 3.4)
LDA	Latent Dirichlet Allocation (Section 2.5.4)
LSA	Latent Semantic Analysis (Section 1.2)
LSI	Latent Semantic Indexing (Section 1.2)
LSTM	Long Short-Term Memory (Section 1.3.2.1)
MIML	Multi-Instance Multi-Label (Section 2.4.2)
MLM	Masked Language Model (Section 1.3.4.2)
MTB	Matching The Blanks (Sections 2.3.7 and 2.5.6)
MUC	Message Understanding Conference (Section C.4)
NLP	Natural Language Processing (Sections 1.2 and 1.3)
NCE	Noise Contrastive Estimation (Section 1.2.1.2)
NER	Named Entity Recognition (Chapter 2)
NIST	National Institute of Standards and Technology (Section C.1)
NMT	Neural Machine Translation (Section 1.3.3)
NYT	New York Times (Section C.5)
OIE	Open Information Extraction (Section 2.5.2)
PCNN	Piecewise Convolutional Neural Network (Section 2.3.6)
PMI	Pointwise Mutual Information (Section 2.3.3)

POS Part Of Speech (Figure 2.4)
RI Rand Index (Section 2.5.1.1)
RNN Recurrent Neural Network (Section 1.3.2)
SVM Support Vector Machine (Section 2.3.5)
SGNS Skip-Gram Negative Sampling (Section 1.2.1)
TF Term Frequency (Section 2.5.3)
VAE Variational AutoEncoder (Section 2.5.5)
WL Weisfeiler–Leman isomorphism test (Section 4.3.5)
WMT Workshop on statistical Machine Translation (Section 1.1)

Notation

Most of this thesis is formatted in one and a half columns, which means that a large right margin is filled with complementary material. This includes figures, tables and algorithms when space allows, but also epigraphs and marginal notes with supplementary details and comments. The titles of important bibliographical references are also given in the margin right of their first mention in the section. Some marginal paragraphs are left unnumbered and provide material about the broadly adjacent passage. When a section seems unclear, we invite the reader to look for additional information in the margin. For example, while relation algebra is introduced in Section 1.4.1, we do not expect most readers to be familiar with its notation. As such, we will systematically provide an interpretation of relation algebra formulae in plain English in unnumbered marginal paragraphs.

Domain of Variables

x	A scalar
$\boldsymbol{x}$	A vector, its elements are indexed x_i
$\boldsymbol{X}$	A matrix, its rows are indexed $\boldsymbol{x}_i$, its elements x_{ij}
$\boldsymbol{\mathsf{X}}$	A (three-way) tensor, indexed $\boldsymbol{X}_i$, $\boldsymbol{x}_{ij}$, x_{ijk}
x	A random variable (sometimes X to avoid confusion)
$\mathbf{x}$	A random vector
$\mathbb{R}$	The set of real numbers
$\mathbb{R}^n$	The set of real-valued vectors of length n
$\mathbb{R}^{n\times m}$	The set of real-valued matrices with n rows and m columns
B^A	The set of functions from A to B, in particular 2^A denotes the power set of A

To describe the set of real-valued vectors with the same number of elements as a set A, we abuse the morphism from the functions $\mathbb{R}^A$ to the vectors $\mathbb{R}^{|A|}$ and simply write $\boldsymbol{x} \in \mathbb{R}^A$ to denote that $\boldsymbol{x}$ is a vector with $|A|$ elements.

Relation Algebra

Relation algebra is described in more detail in Section 1.4.1.

$\mathbf{0}$	Empty relation
$\mathbf{1}$	Complete relation
$\boldsymbol{I}$	Identity relation
$\bar{r}$	Complementary relation
$\breve{r}$	Converse relation (reversed orientation), when applied to a surface form: $\overset{\frown}{\textit{born in}}$
$\bullet$	Relation composition

Probability and Information Theory

$P(\mathrm{x})$, $Q(\mathrm{x})$	Probability distribution over x, by default we heavily overload P (as is customary), when confusion is possible we disambiguate by using Q
$\hat{P}(\mathrm{x})$	Empirical distribution over x (as defined by the dataset)
$\mathrm{x} \perp\!\!\!\perp \mathrm{y} \mid \mathrm{z}$	Conditional independence of x and y given z
$\mathrm{x} \not\perp\!\!\!\perp \mathrm{y}$	x and y are not independent
$\mathcal{U}(X)$	Uniform distribution over the set X

$\mathcal{N}(\mu, \sigma^2)$	Normal distribution of mean μ and variance σ^2 (also used for the multivariate case)
$\mathrm{H}(\mathrm{x})$	Shannon entropy of the random variable x, $\mathrm{H}(\mathrm{x}, \mathrm{y})$ denotes the joint entropy
$\mathrm{H}(\mathrm{x} \mid \mathrm{y})$	Conditional entropy of x given y
$\mathrm{H}_Q(P)$	Cross-entropy of P relative to Q
$\mathrm{I}(\mathrm{x}; \mathrm{y})$	Mutual information of x and y
$\mathrm{pmi}(x, y)$	Pointwise mutual information of events x and y
$\mathrm{D}_{\text{KL}}(P \parallel Q)$	Kullback–Leibler divergence from Q to P
$\mathrm{D}_{\text{JSD}}(P \parallel Q)$	Jensen–Shannon divergence between P and Q
$W_1(P, Q)$	1-Wasserstein distance between P and Q

Machine Learning

$\sigma(x)$	Logistic sigmoid $\sigma(x) = 1 \mathbin{/} (1 + \exp(-x))$
$\mathrm{ReLU}(x)$	Rectified linear unit $\mathrm{ReLU}(x) = \max(0, x)$, we use $\mathrm{ReLU}_{\LEFTcircle}$ to refer to the ReLU activation applied to half of the units (see Section 1.3.3.2)
$\mathcal{L}$	Loss (to be minimized)
J	Objective (to be maximized)
$\overrightarrow{F_1}$, $\overleftrightarrow{F_1}$, $\overleftarrow{F_1}$	Directed, undirected and half-directed F_1 measures (see Section 2.3.1)

Graph Operations

$\varepsilon_1(a)$	Source vertex of the arc a
$\varepsilon_2(a)$	Target vertex of the arc a
$\rho(a)$	Relation conveyed by the arc a
$\varsigma(a)$	Sentence corresponding to the arc a
$N(e)$	Vertices neighboring the vertex e
$\mathcal{I}(e)$	Arcs incident to the vertex e
$\mathcal{N}(a)$	Arcs neighboring the arc a

Other Operations

$\odot$	Element-wise (Hadamard) product
$*$	Convolution
$\bowtie$	Natural join
$\times_A$	Pullback with common codomain A
$\delta_{i,j}$	Kronecker's delta, 1 if $i = j$, 0 otherwise

Introduction

The world is endowed with a structure, which enables us to understand it. This structure is most apparent through repetitions of sensory experiences. Sometimes, we can see a cat, then another cat. Entities emerge from the repetition of catness we experienced. From time to time, we can also observe a cat *inside* a cardboard box or a person *inside* a room. Relations are the explanatory device underlying this second kind of repetition. A relation governs an interaction between two or more objects. We assume an *inside* relation exists because we repeatedly experienced the same interaction between a container and its content. The twentieth century saw the rise of structuralism, which regarded the interrelations of phenomena as more enlightening than the study of phenomena in isolation. In other words, we might better understand what a cat is by studying its relationships to other entities instead than by listing the characteristics of catness. From this point of view, the concept of relation is crucial to our understanding of the world.

Relations—albeit in a more restrictive sense—are one of Aristotle's ten *praedicamenta*, the categories of objects of human apprehension (Gracia and Newton 2016).

The Cheshire Cat from Tenniel (1889) provides you with an experience of catness.

Natural languages capture the underlying structure of these repetitions through a process we do not fully understand. One of the endeavors of artificial intelligence, called natural-language understanding, is to mimic this process with definite algorithms. Since the aforementioned goal is still elusive, we strive to model only parts of this process. This thesis, consequent to the structuralist perspective, focuses on extracting relations conveyed by natural language. Assuming natural language is representative of the underlying structure of sensory experiences,[1] we should be able to capture relations through the exploitation of repetitions alone—i.e. in an unsupervised fashion.

Extracting relations can help better our understanding of how languages work. For example, whether languages can be understood through a small amount of data is still a somewhat open question in linguistics. The poverty of the stimulus argument states that children should not be able to acquire proficiency from being exposed to so little data. It is one of the major arguments in favor of the controversial universal grammar theory. Capturing relations from nothing more than a small number of natural language utterances would be a step towards disproving the poverty of the stimulus claim.

[1] The repetitions of sensory experiences and words need not be alike. We are only concerned with the possibility of resolving references here. Even though our experiences of trees are more often than not accompanied with experiences of bark, the words "tree" and "bark" do not co-occur as often in natural language utterances. However, their meronymic relationship is understandable both through experiences of trees and inter alia through the use of the preposition "of" in textual mentions of barks.

This kind of incentive for tackling the relation extraction problem stems from an *episteme*[2] endeavor. However, most of the traction for this problem stems from a *techne*[3] undertaking. The end goal is to build a system with real-world applications. Under this perspective, the point of artificial intelligence is to replace or assist humans on specific tasks. Most tasks of interest necessitate some form of technical knowledge (e.g. diagnosing a disease requires knowledge of the relationship between symptoms and diseases). The principal vector of knowledge is language (e.g. through education). Thus, knowledge acquisition from natural language is fundamental for systems purposing to have such applications.

[2] From the Ancient Greek ἐπιστήμη: knowledge, know-how.

[3] From the Ancient Greek τέχνη: craft, art.

For an analysis of the real-world impact of systems extracting knowledge from text, refer to Alex et al. (2008). Their article shows that human curators can use a machine learning system to better extract a set of protein–protein interactions from biomedical literature. This is clearly a *techne* endeavor: the protein–protein interactions are not new knowledge, they are already published; however, the system improves the work of the human operator.

Alex et al., "Assisted curation: does text mining really help?" PSB 2008

> *Once the theory of meaning is sharply separated from the theory of reference, it is a short step to recognizing as the business of the theory of meaning simply the synonymy of linguistic forms and the analyticity of statements; meanings themselves, as obscure intermediary entities, may well be abandoned.*
>
> — Willard Van Orman Quine, "Main Trends in Recent Philosophy: Two Dogmas of Empiricism" (1951)

This example of application is revealing of the larger problem of information explosion. The quantity of published information has grown relentlessly throughout the last decades. Machine learning can be used to filter or aggregate this large amount of data. In this case, the object of interest is not the text in itself but the conveyed semantic, its meaning. This begs the question: how to define the meaning we are seeking to process? Indeed, foundational theories of meaning are the object of much discussion in the philosophy community (Speaks 2021). While some skeptics, like Quine, do not recognize meaning as a concept of interest, they reckon that a minimal description of meaning should at least encompass the recognition of synonymy. This follows from the above discussion about the recognition of repetitions: if 🐇 is a repetition of 🐇, we should be able to say that 🐇 and 🐇 are synonymous. In practice, this implies that we ought to be able to extract classes of linguistic forms with the same meaning or referent—the difference between the two is not relevant to our problem.

Paris (Q162121) is neither capital of France, nor prince of Troy, it is the genus of the true lover's knot plant. The capital of France would be Paris (Q90) and the prince of Troy, son of Priam, Paris (Q167646). Illustration from Redouté (1802).

While the above discussion of meaning is essential to define our objects of interest, relations, it is important to note that we work on language; we want to extract relations from language, not from repetitions of abstract entities. Yet, the mapping between linguistic signifiers and their meaning is not bijective. We can distinguish two kinds of misalignment between the two: either two expressions refer to the same object (synonymy), or the same expression refers to different objects depending on the context in which it appears (homonymy). The first variety of misalignment is the most common one, especially at the sentence level. For example, "Paris is the capital of France" and "the capital of France is Paris" convey the same meaning despite having different written and spoken forms. On the other hand, the second kind is principally visible at the word level. For example, the preposition "from" in the phrases "retinopathy from diabetes" and "Bellerophon from Corinth" conveys either a *has effect* relationship or a *birthplace* one. To distinguish these two uses of "from," we can use relation identifiers such as P1542 for *has effect* and P19 for *birthplace*. An example with entity identifiers—which purpose to uniquely identify entity concepts—is provided in the margin of page xx.

Throughout this thesis, we will be using Wikidata identifiers (`https://www.wikidata.org`) to index entities and relations. Entities identifiers start with `Q`, while relation identifiers start with `P`. For example, Q35120 is an entity.

While the preceding discussion makes it seems as if all objects can fit nicely into clearly defined concepts, in practice, this is far from the truth. Early in the knowledge-representation literature, Brachman (1983) remarked the difficulty to clearly define even seemingly simple relations such as *instance of* (P31). This problem ensues from the assumption that synonymy is transitive, and therefore, induces equivalence classes. This assumption is fairly natural since it already applies to the link between language and its references: even though two cats might be very unlike one another, we still group them under the same signifier. However, language is flexible. When trying to capture the entity "cat," it is not entirely clear whether we should group "a cat with the body of a cherry pop tart" with regular experiences of catness.[4] To circumvent this issue, some recent works (Han et al. 2018) on the relation extraction problem define synonymy as a continuous intransitive association. Instead of grouping linguistic forms into clear-cut classes with a single meaning, they extract a similarity function defining how similar two objects are.

[4] The reader who would describe this as a cat is invited to replace various body parts of this imaginary cat with food items until they stop experiencing catness.

Now that we have conceptualized our problem, let us focus on our proposed technical approach. First, to summarize, this thesis focus on unsupervised relation extraction from text.[5] Since relations are objects capturing the interactions between entities, our task is to find the relation linking two given entities in a piece of text. For example, in the three following samples where entities are underlined:

[5] We use text as it is the most definite and easy-to-process rendition of language.

> <u>Megrez</u>$_{e_1}$ is a star in the northern circumpolar constellation of <u>Ursa Major</u>$_{e_2}$.
>
> <u>Posidonius</u>$_{e_1}$ was a Greek philosopher, astronomer, historian, mathematician, and teacher native to <u>Apamea, Syria</u>$_{e_2}$.
>
> <u>Hipparchus</u>$_{e_1}$ was born in <u>Nicaea, Bithynia</u>$_{e_2}$, and probably died on the island of Rhodes, Greece.

we wish to find that the last two sentences convey the same relation—in this case, e_1 *born in* e_2 (P19)—or at the very least, following the discussion in the preceding paragraph about the difficulty of defining clear relation classes, we wish to find that the relations conveyed by the last two samples are closer to each other than the one conveyed by the first sample. We propound that this can be performed by machine learning algorithms. In particular, we study how to approach this task using deep learning. While

Ariadne waking on the shore of Naxos where she was abandoned, wall painting from Herculaneum in the collection of the British Museum (100 BCE–100 CE). The ship in the distance can be identified as the ship of Theseus, for now. Depending on the philosophical view of the reader (Q1050837), its identity as the ship of Theseus might not linger for long.

relation extraction can be tackled as a standard supervised classification problem, labeling a dataset with precise relations is a tedious task, especially with technical documents such as the biomedical literature studied by Alex et al. (2008). Another problem commonly encountered by annotators is the question of applicability of a relation, for example, should "the $\underline{\text{country}}_{e_1}$'s founding $\underline{\text{father}}_{e_2}$" be labeled with the *product–producer* relation?[6] We now discuss how deep learning became the most promising technique to tackle natural language processing problems.

[6] The annotator of this sentence piece in the SemEval 2010 Task 8 dataset (Section C.6) decided that it does convey the *product–producer* relation. The difficulty of applying a definition is an additional argument in favor of similarity-function-based approaches over classification approaches.

The primary subject matter of the relation extraction problem is language. Natural language processing (NLP) was already a prominent research interest in the early years of artificial intelligence. This can be seen from the *episteme* viewpoint in the seminal paper of Turing (1950). This paper proposes mastery of language as evidence of intelligence, in what is now known as the Turing test. Language was also a subject of interest for *techne* objectives. In January 1954, the Georgetown–IBM experiment tried to demonstrate the possibility of translating Russian into English using computers (Dostert 1955). The experiment showcased the translation of sixty sentences using a bilingual dictionary to translate words individually and six kinds of grammatical rules to reorder tokens as needed. Initial experiments created an expectation buildup, which was followed by an unavoidable disappointment, resulting in an "AI winter" where research fundings were restricted. While translating word-by-word is somewhat easy in most cases, translating whole sentences is a lot harder. Scaling up the set of grammatical rules in the Georgetown–IBM experiment proved impractical. This limitation was not a technical one. With the improvement of computing machinery, more rules could have easily been encoded. One of the issues identified at the time was the commonsense knowledge problem (McCarthy 1959). In order to translate or, more generally, process a sentence, it needs to be understood in the context of the world in which it was uttered. Simple rewriting rules cannot capture this process.[7] In order to handle whole sentences, a paradigm shift was necessary.

Turing, "Computing Machinery and Intelligence" Mind 1950

> *Five, perhaps three years hence, interlingual meaning conversion by electronic process in important functional areas of several languages may well be an accomplished fact.*
>
> — Leon Dostert, "701 translator" IBM press release (1954)

[7] Furthermore, grammar is still an active area of research. We do not perfectly understand the underlying reality captured by most words and are thus unable to write down complete formal rules for their usages. For example, Tyler and Evans (2001) is a 43 pages cognitive linguistics paper attempting to explain the various uses of the English preposition "over." This is one of the arguments for unsupervised approaches; we should avoid hand-labeled datasets if we want to outperform the human annotators.

A first shift occurred in the 1990s with the advent of statistical NLP (S. Abney 1996). This evolution can be partly attributed to the increase of computational power, but also to the progressive abandon of essentialist linguistics precepts[8] in favor of distributionalist ones. Instead of relying on human experts to input a set of rules, statistical approaches leveraged the repetitions in large text corpora to infer these rules automatically. Therefore, this progression can also be seen as a transition away from symbolic artificial intelligence models and towards statistical ones. Coincidently, the relation extraction task was formalized at this time. And while the earliest approaches were based on symbolic models using handwritten rules, statistical methods quickly became the norm after the 1990s. However, statistical NLP models still relied on linguistic knowledge. The relation

[8] Noam Chomsky, one of the most—if not the most—prominent essentialist linguists, considers that manipulating probabilities of text excerpt is not the way to acquire a better understanding of language. Following the success of statistical approaches, he only recognized statistical NLP as a *techne* achievement. For an answer to this position, see S. Abney (1996) and Norvig (2011).

extraction systems were usually split into a first phase of hand-specified linguistic features extraction and a second phase where a relation was predicted based on these features using shallow statistical models.

A second shift occurred in the 2010s when deep learning approaches erased the split between feature extraction and prediction. Deep learning models are trained to directly process raw data, in our case text excerpts. To achieve this feat, neural networks able to approximate any function are used. However, the downside of these models is that they usually require large amounts of labeled data to be trained. This is a particularly salient problem throughout this thesis since we deal with an unsupervised problem. As the latest and most efficient technique available, deep learning proved to be a natural choice to tackle relation extraction. However, this natural evolution came with serious complications that we try to address in this manuscript.

The evolution of unsupervised relation extraction methods closely follows the one of NLP methods described above. The first deep learning approach was the one of Marcheggiani and Titov (2016). However, only part of their model relied on deep learning techniques, the extraction of features was still done manually. The reason why feature extraction could not be done automatically as is standard in deep learning approaches is closely related to the unsupervised nature of the problem. Our first contribution is to propose a technique to enable the training of unsupervised fully-deep learning relation extraction approaches. Afterward, different ways to tackle the relation extraction task emerged. First, recent approaches use a softer definition of relations by extracting a similarity function instead of a classifier. Second, they consider a broader context: instead of processing each sentence individually, the global consistency of extracted relations is considered. However, this second approach was mostly limited to the supervised setting, with limited use in the unsupervised setting. Our second contribution concerns using this broader context for unsupervised relation extraction, in particular for approaches defining a similarity function. During the preparation of the thesis, we also published an article on multimodal semantic role labeling with Syrielle Montariol and her team (Montariol et al. 2022); since it is somewhat unrelated to unsupervised relation extraction, we do not include it in this thesis.

We now describe the organization of the thesis. Chapter 1 provides the necessary background for using deep learning to tackle the relation extraction problem. In particular, we focus on the concept of distributed representation, first of language, then of entities and relations. Chapter 2 formalizes the relation extraction task and presents the evaluation framework and relevant related works. This chapter focuses first on supervised relation extraction using local information only, then on aggregate extraction, which exploits repetitions more directly, before delving into unsu-

「白馬非馬」

“ *White horse is not horse.*
— “Gongsun Longzi” Chapter 2 (circa 300 BCE)

A well-known paradox in early Chinese philosophy illustrating the difficulty of clearly defining the meaning conveyed by natural languages. This paradox can be resolved by disambiguating the word “horse.” Does it refers to the “whole of all horse kind” (the mereological view) or to “horseness” (the Platonic view)? The mereological interpretation was famously—and controversly—introduced by Hansen (1983), see Fraser (2007) for a discussion of early Chinese ontological views of language.

Frontispiece of the OuCuiPian Library by Chevalier (1990). A different kind of cooking with letters.

Syrielle Montariol,* Étienne Simon,* Arij Riabi, Djamé Seddah. “Fine-tuning and Sampling Strategies for Multimodal Role Labeling of Entities under Class Imbalance” CONSTRAINT 2022

* Equal contributions

pervised relation extraction. In Chapter 3, we propose a solution to train deep relation extraction models in an unsupervised fashion. The problem we tackle is a stability problem between a powerful universal approximator and a weak supervision signal transpiring through the repetitions in the data. This chapter was the object of a publication at ACL (Simon et al. 2019). Chapter 4 explores the methods to exploit the structure of the data more directly through the use of graph-based models. In particular, we draw parallels with the Weisfeiler–Leman isomorphism test to design new methods using topological (dataset-level) and linguistic (sentence-level) features jointly. Appendix A contains the state-mandated thesis summary in French. The other appendices provide valuable information that can be used as references. We strongly encourage the reader to refer to them for additional details on the datasets (Appendix C), but even more so for the list of assumptions made by relation extraction models (Appendix B). These modeling hypotheses are central to the design of unsupervised approaches. In addition to their definition and reference to the introducing section, Appendix B provides counterexamples, which might help the reader understand the nature of these assumptions.

Étienne Simon, Vincent Guigue, Benjamin Piwowarski. "Unsupervised Information Extraction: Regularizing Discriminative Approaches with Relation Distribution Losses" ACL 2019

The work presented in Chapter 4 still needs to be polished with more experimental work and is yet unpublished at the time of writing.

Chapter 1

Context: Distributed Representations

> *Meaning is what essence becomes when it is divorced from the object of reference and wedded to the word.*
>
> — Willard Van Orman Quine, "Main Trends in Recent Philosophy: Two Dogmas of Empiricism" (1951)

Quine was skeptical that facts about the meanings of linguistic expressions existed, for a critical response to his position see Soames (1997).

> 「論貴是而不務華」
>
> *In scientific discourse what matters are the solid facts of a matter, not elegance.*
>
> — Wang Chong, "Lunheng" Chapter 85 (circa. 80)

Adapted from the translation of Harbsmeier (1989), Chong promotes truth over elegance despite the influence of early Chinese skepticism.

Language conveys meaning. Thus, it should be possible to explicitly map a text to its semantic content. The research reported in this thesis seeks to algorithmically extract meaning conveyed by language using deep learning techniques from the information extraction and natural language processing (NLP) fields. We focus on the task of relation extraction, in which we seek to extract the semantic relation conveyed by a sentence. For example, given the sentence "Paris is the capital of France," we seek to extract the relation "*capital of*." To build a formal representation of relations, we use knowledge bases. In their simplest form, knowledge bases encode knowledge as a set of facts, which take the form (entity, relation, entity) such as (Paris, *capital of*, France). Like natural languages, knowledge bases purpose to convey meaning[9] but in a structure that is readily manipulable by algorithms. However, most knowledge—like this thesis—comes in the form of text. There lies the usefulness of the relation extraction task on which we focus. By "translating" natural language into knowledge bases, we seek to make more knowledge available to algorithms.

In this chapter, we focus on the two kinds of data we deal with in this thesis, namely text and knowledge bases. Subsequent chapters will deal with the extraction of knowledge base facts from text. In Section 1.1, we begin by positioning this task within the larger historical context by focusing on how the fields of machine learning, NLP and information extraction developed. Before delving into the specific algorithms for relation extraction, we must first define how to process language and how to represent semantic information in a way that can be manipulated by machine learning algorithms. In particular, we seek to obtain a *distributed representation*—which we define in the next section—of both language and knowledge bases since deep learning algorithms cannot directly work with non-distributed representations. We first inspect the representation of words in Section 1.2 before exploring how to process whole sentences in Section 1.3. Finally, Section 1.4 focuses on knowledge bases by first giving a formal definition before studying methods for extracting distributed representations from them.

[9] Knowledge bases usually focus on knowledge which can be seen as a subset of all possible meanings. For example, facts like (I, *want*, ice cream) are not usually encoded in knowledge bases. However, they theoretically could. To be precise, throughout this thesis we'll be using knowledge bases in two ways:

- as a basic theoretical structured representation of meaning,
- as a practical datasets to evaluate algorithms on.

This means that algorithms tested on existing knowledge bases are only tested on a subset of possible meanings. However, when we discuss the representation of knowledge base facts, note that this can be generalized to any meaningful facts expressible in the knowledge base framework.

1.1 Historical Development

In this section, we expose the rationale for applying deep learning to relation extraction, how the related fields appeared and why the task is relevant. Since algorithms were first given to train generic deep neural networks (Glorot et al. 2011; Geoffrey E. Hinton et al. 2006), most problems tackled by machine learning can now be approached with deep learning methods. Over the last few years, deep learning has been very successful in a variety of tasks such as image classification (Krizhevsky et al. 2012), machine translation (Cho et al. 2014), audio synthesis (van den Oord et al. 2016), etc. This is why it is not surprising that deep learning is now applied to more tasks traditionally tackled by other machine learning methods, such as in this thesis, where we apply it to relation extraction.

From a historical point of view, machine learning—and hence deep learning—are deeply anchored in *empiricism*. Empiricism is the epistemological paradigm in which knowledge is anchored in sensory experiences of the world, which are called empirical evidence. This is not to say that there are no theoretical arguments motivating the use of certain machine learning methods; the universal approximation theorems (Cybenko 1989; Leshno et al. 1993) can be seen as a theoretical argument for deep learning. But in the end, a machine learning method draws its legitimacy from the observation that they perform strongly on a real dataset. This is in stark contrast to the rationalist paradigm, which posits that knowledge comes primarily from reason.

This strong leaning on empiricism can also be seen in NLP. NLP comes from the *externalist* approach to linguistic theorizing, focusing its analyses on actual utterances. A linguistic tool that externalists often avoid while being widely used by other schools is elicitation through prospective questioning: "Is this sentence grammatical?" Externalists consider that language is acquired through distributional properties of words and other constituents;[10] and study these properties by collecting corpora of naturally occurring utterances. The associated school of structural linguistics inscribes itself into the broader view of *structuralism*, the belief that phenomena are intelligible through a concept of structure that connects them together, the focus being more on these interrelations instead of each individual object. In the case of linguistics, this view was pioneered by Ferdinand de Saussure which stated in its course in general linguistics:

> Language is a system whose parts can and must all be considered in their synchronic[11] solidarity.
>
> — Ferdinand de Saussure, *Cours de linguistique générale* (1916)

This train of thought gave rise to *distributionalism* whose ideas are best illustrated by the distributional hypothesis stated in Harris (1954):

Distributional Hypothesis: *Words that occur in similar contexts convey similar meanings.*

This can be pushed further by stating that a word is solely characterized by the context in which it appears.

On the artificial intelligence side, deep learning is usually compared to symbolic approaches. The distinction originates in the way information is represented by the system. In the symbolic approach, information is carried by strongly structured representations in which a concept is usually associated with a single entity, such as a variable in a formula or in a probabilistic graphical model. On the other hand, deep learning uses distributed representations in which there is a many-to-many relationship between concepts and neurons; each concept is represented by many neurons, and each neuron represents many concepts. The idea that mental

[10] In other words, language is acquired by observing empirical co-occurrences: where words go and where they don't in actual utterances tell us where they can go and where they can't.

> “ *La langue est un système dont toutes les parties peuvent et doivent être considérées dans leur solidarité synchronique.*
>
> — Ferdinand de Saussure, *Cours de linguistique générale* (1916)

[11] Saussure makes a distinction between synchronic—at a certain point in time—and diachronic—changing over time—analyses. This does not mean that the meaning of a word is not influenced by its history, but that this influence is entirely captured by the relations of the word with others at the present time and that conditioned on these relations, the current meaning of the word is independent of its past meaning.

phenomena can be represented using this paradigm is known as *connectionism*. One particular argument in favor of connectionism is the ability to degrade gracefully: deleting a unit in a symbolic representation equates to deleting a concept, while deleting a unit in a distributed representation merely lowers the precision with which concepts are defined. Note that connectionism is not necessarily incompatible with a symbolic theory of cognition. Distributed representations can be seen as a low-level explanation of cognition, while from this point of view, symbolic representation is a high-level interpretation encoded by distributed representations.[12]

[12] This view on the relation between distributed and symbolic representations can be seen in the early neural networks literature as can be seen in Geoffrey E Hinton (1986), which is often cited for its formalization of the backpropagation algorithm. More recently, Greff et al. (2020) investigate the binding problem between symbols and distributed representations.

Furthermore, we can make a distinction on how structured is the kind of data used. In this thesis, we will especially focus on the relationship between unstructured text[13] and structured data (in the form of knowledge bases). To give a sense of this difference, compare the following text from the Paris Wikipedia page to facts from the Wikidata knowledge base:

[13] Of course, language does have a structure. We do not deny the existence of grammar but merely state that text is less structured than other structures studied in this chapter (see Section 1.4).

Paris is the capital and most populous city of France. The City of Paris is the centre and seat of government of the region and province of Île-de-France.	Paris *capital of* France Paris *located in the administrative territorial entity* Île-de-France

We use *slanted text* to indicate a relational surface form such as "*capital of*" in the fact "Paris *capital of* France."

Through this example, we see that both natural languages and knowledge bases encode meaning. To talk about what they encode, we assume the existence of a semantic space containing all possible meanings. We do not assume any theory of meaning used to define this space; this allows us to stay neutral on whether language is ontologically prior to propositional attitudes and its link with reality or semantically evaluable mental states. In the same way that different natural languages are different methods to address this semantic space, knowledge bases seek to refer to the same semantic space[14] with an extremely rigid grammar.

[14] Strictly speaking, practical knowledge bases only seek to index a subset of this space, see note 9 in the margin of page 25.

Both natural language and knowledge bases are discrete systems. For both these systems, we can use the distributional hypothesis to obtain continuous distributed representations. These representations purpose to capture the semantic as a simple topological space such as a Euclidean vector space where distance encodes dissimilarity, as shown in Figure 1.1. Moreover, using a differentiable manifold allows us to train these representations through backpropagation using neural architectures.

The question of how to process texts algorithmically has evolved over the last fifty years. Language being conveyed through symbolic representations, it is quite natural for us to manipulate them. As such, early machine learning models strongly relied on them. For a long time, symbolic approaches had an empirical advantage: they worked better. However, in the last few years, distributed representations have shown unyielding results, and most tasks are now tackled with deep learning using distributed representations. As an example, this can be seen in the machine translation task. Early models from the 1950s onward were rule-based. Starting in the 1990s, statistical approaches were used, first using statistics of words then of phrases. Looking at the Workshop on statistical machine translation (WMT): at the beginning of the last decade, no neural approaches were used and the report (Callison-Burch et al. 2010) deplored the disappearance of rule-based systems, at the end of the decade, most systems were based on distributed representations (Barrault et al. 2020).[15] While this transition occurred in NLP, knowledge representation has been a stronghold of symbolic approaches until very recently. The research reported in this thesis

This transition from rule-based models to statistical models to neural network models can also be seen in relation extraction with Hearst (1992, symbolic rule-based, Section 2.2.1), SIFT (1998, symbolic statistical, Section 2.3.4) and PCNN (2015, distributed neural, Section 2.3.6).

[15] To be more precise, most models use transformers which are a kind of neural network introduced in Section 1.3.4.

aims to develop the distributed approach to knowledge representation for the task of relation extraction. In the remainder of this chapter, we first report the distributed approaches to NLP, which showcased state-of-the-art results for the last decade, before presenting a structured symbolic representation, knowledge bases, and some methods to obtain distributed representations from them.

1.2 Distributed Representation of Words

Natural language processing (NLP) deals with the automatic manipulation of natural language by algorithms. Nowadays, a large pan of NLP concerns itself with the question of how to obtain good distributed representations from textual inputs. What constitutes a good representation may vary, but it is usually measured by performance on a task of interest. Natural language inputs present themselves as tokens or sequences of tokens, usually in the form of words stringed together into sentences. The goal is then to map these sequences of symbolic units to distributed representations. This section and the next present several methods designed to achieve this goal which have become ubiquitous in NLP research. We first describe how to obtain good representations of words—or of smaller semantic units in Section 1.2.3—before studying how to use these representations to process whole sentences in Section 1.3.

Given a vocabulary, that is a set of words $V = \{\text{a}, \text{aardvark}, \text{aback}, \dots\}$, our goal is to map each word $w \in V$ to an embedding $u_w \in \mathbb{R}^d$ where d is a hyperparameter. An example of an embedding space is given in Figure 1.1. One of the early methods to embed words like this is latent semantic analysis (LSA, Dumais et al. 1988). Interestingly, LSA was popularized by the information retrieval field under the name latent semantic indexing (LSI). The basis of LSA is a document–term matrix indicating how many times a word appears in a document. A naive approach would be to take the rows of this matrix; we would obtain a vector representation of each word, the dimension d of these embeddings would be the number of documents. The similarity of two words is then evaluated by taking the cosine similarity of the associated vectors; in the simple case described above, this value would be high if the two words often appear together in the same documents and low otherwise. We can already see that this representation is distributed since each document makes up a small fraction of the representation of the words it contains. However, this approach is not practical, as either d is too large, or the representations obtained tend to be noisy (when the number of documents is relatively small). So LSA goes one step further and builds a low-rank approximation of this matrix such that d can be chosen as small as we want. This basic idea of modeling word co-occurrences forms the basis behind most word embedding techniques.

In contrast, a symbolic representation of words would simply map each word to an index $V \to \{1, \dots, |V|\}$.

Dumais et al., "Using latent semantic analysis to improve access to textual information" SIGCHI 1988

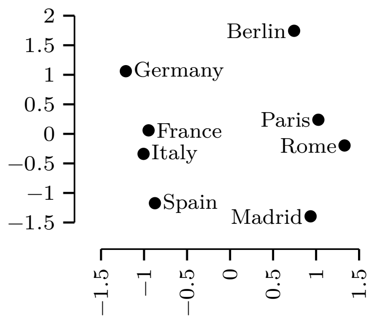


Figure 1.1: Selected word2vec embeddings of dimension $d = 300$, projected into two dimensions using PCA (explained variance ratio 27.6%+25.4%). The representations encode a strong separation between countries and capitals. Furthermore, the relative position of each country with respect to its associated capital is somewhat similar.

In this section, we focus on the representation of words, yet most NLP tasks need to process longer chunks of text; this will be the focus of Section 1.3. We center our overview of word representations on word2vec in Section 1.2.1. With the advent of deep learning, word2vec has been the most ubiquitous word embedding technique. Additionally, it introduced negative sampling, a technique that we make use of in Chapter 3. Section 1.2.2 introduces the notion of language model, which is central to several representation extraction techniques in NLP; we also present several alternatives to word2vec used before the transition to sentence-level approaches of Section 1.3.4. Finally, while models presented in this section are focused on words, smaller semantic units can similarly be used. This is especially needed for languages in which words have a complex internal structure, but it can also be applied to English. Section 1.2.3 will explore alternative levels at which we can apply methods from Sections 1.2.1 and 1.2.2.

1.2.1 Word2vec

Word2vec (Mikolov et al. 2013a,b) is one of the first NLP models widely used for the representations it produces. As its name implies, word2vec outputs word representations; however, its general framework can be used on other kinds of tokens. Word2vec relies strongly on the distributional hypothesis: its goal is to model the context of a word to produce a representation of the word itself, a technique which was pioneered by Bengio et al. (2003). Several variants of the word2vec model exist, but for the sake of conciseness, this section focuses on the skip-gram with negative sampling (SGNS) approach.

Mikolov et al., "Distributed Representations of Words and Phrases and their Compositionality" NeurIPS 2013

Bengio et al., "A Neural Probabilistic Language Model" JMLR 2003

1.2.1.1 Skip-gram

Given a word, the idea behind skip-gram is to model its context.[16] The probability of a word $c \in V$ to appear in the context of a word $w \in V$ is modeled by the following softmax:

$$P(c \mid w) = \frac{\exp(\boldsymbol{u}_w^\top \boldsymbol{u}'_c)}{\sum_{c' \in V} \exp(\boldsymbol{u}_w^\top \boldsymbol{u}'_{c'})} \tag{1.1}$$

where V is the vocabulary, and $\boldsymbol{U}, \boldsymbol{U}' \in \mathbb{R}^{V \times d}$ are the model parameters assigning a vector representation to all words in the vocabulary. The rows of these parameters $\boldsymbol{u}_w$ and $\boldsymbol{u}'_w$ are what is of interest when word2vec is used for transfer learning. Once the model has been trained, $\boldsymbol{u}_w$ can be used as a distributed representation for w, capturing its associated semantics. See Figure 1.1 for an example of extracted vectors.

[16] The context of a word w is defined as all words appearing in a fixed-size window around w in the text. In the case of word2vec, this window is of size five in both directions.

Here, we omit the conditioning on the parameters. More formally, $P(c \mid w)$ should be written $P(c \mid w; \boldsymbol{U}, \boldsymbol{U}')$.

1.2.1.2 Noise Contrastive Estimation

Evaluating Equation 1.1 is quite expensive since the normalization term involves all the words in the vocabulary. Noise Contrastive Estimation (NCE, Gutmann and Hyvärinen 2010) is a training method that removes the need to compute the partition function of probabilistic models explicitly. To achieve this, NCE reframes the model as a binary classification problem by modeling the probability that a data point—in word2vec's case a word-context pair—comes from the observed dataset $P(\mathrm{D} = 1 \mid w, c)$. This probability is contrasted with k samples from a noise distribution following the unigram distribution $\hat{P}(\mathrm{W})$, that is the empirical word frequency.[17] This translate to $P(c \mid \mathrm{D} = 1, w) = \hat{P}(c \mid w)$ and $P(c \mid \mathrm{D} = 0, w) = \hat{P}(\mathrm{W} = c)$. Using the prior $P(\mathrm{D} = 0) = \frac{k}{k+1}$, the posterior can be expressed as:

$$P(\mathrm{D} = 1 \mid w, c) = \frac{\hat{P}(c \mid w)}{\hat{P}(c \mid w) + k\hat{P}(c)}. \tag{1.2}$$

Restating Equation 1.1 as $P(c \mid w) = \exp(\boldsymbol{u}_w^\top \boldsymbol{u}'_c) \times \gamma_w$ and treating γ_w as another model parameter, NCE allows us to train $\boldsymbol{U}$ and $\boldsymbol{U}'$ without

Gutmann and Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models" AISTATS 2010

We use $\hat{P}$ to refer to empirical distributions, whereas P denotes a modeled probability. For example, $\hat{P}(c \mid w)$ is the actual frequency of the word $c \in V$ in the context of $w \in V$. While $P(c \mid w)$ is the probability word2vec assigns to a given pair $(c, w) \in V^2$.

[17] Word2vec actually scales this distribution and uses various other tricks to lessen the effect of frequent words, refer to Mikolov et al. (2013b) for details.

computing the denominator of Equation 1.1. Furthermore, estimating γ_w is not even necessary, since Mnih and Teh (2012) showed that using $\gamma_w = 1$ for all w works well in practice. The final objective maximised by NCE is the log-likelihood of the classification data:

$$J_{\text{NCE}}(w, c) = \log P(\mathrm{D} = 1 \mid w, c) + \sum_{i=1}^{k} \mathop{\mathbb{E}}_{c'_i \sim P(\mathrm{W})} \left[\log P(\mathrm{D} = 0 \mid w, c'_i)\right]. \tag{1.3}$$

Gutmann and Hyvärinen (2010) showed that optimizing J_{NCE} is equivalent to maximizing the log-likelihood using Equation 1.1 under some reasonable assumptions.

1.2.1.3 Negative Sampling

However, SGNS uses a different approximation of Equation 1.1 called negative sampling. The difference is mainly visible in the expression of the objective which simplifies to:

$$J_{\text{NEG}}(w, c) = \log \sigma(\boldsymbol{u}_w^{\mathsf{T}} \boldsymbol{u}'_c) + \sum_{i=1}^{k} \mathop{\mathbb{E}}_{c'_i \sim P(\mathrm{W})} \left[\log \sigma(-\boldsymbol{u}_w^{\mathsf{T}} \boldsymbol{u}'_{c'_i})\right]. \tag{1.4}$$

This can be shown to be similar to NCE, where Equation 1.2 is instead replaced by the following posterior:

$$P(\mathrm{D} = 1 \mid w, c) = \frac{\hat{P}(c \mid w)}{\hat{P}(c \mid w) + 1}. \tag{1.5}$$

Optimizing the objective of Equation 1.4 is not equivalent to maximizing the log-likelihood of the language model. But even though this is not an approximation of the softmax of Equation 1.1, this method has proven to be quite effective at producing good word representations. Levy and Goldberg (2014) explain the effectiveness of word2vec by showing that SGNS can be interpreted as factoring the pointwise mutual information (PMI) matrix between words and contexts. This led to the emergence of GloVe (Pennington et al. 2014), which produces word embeddings by directly factorizing the PMI matrix.

Levy and Goldberg, "Neural Word Embedding as Implicit Matrix Factorization" NeurIPS 2014

The negative sampling algorithm is one of the main contributions of word2vec; it can be used outside NLP to optimize softmax over large domains. In particular, we make use of negative sampling to approximate a softmax over a large number of entities in Chapter 3. Furthermore, even though it was initially presented on words, the algorithm can be used on other kinds of tokens, as we will see in Section 1.2.3.

1.2.2 Language Modeling for Word Representation

Word2vec is part of a large class of algorithms that seek to learn word representation from raw text. More precisely, to obtain distributed representations of natural language inputs, most modern approaches rely on language models. A language model specifies a probability distribution over sequences of tokens $P(w_1, \ldots, w_m)$. The tokens $\boldsymbol{w}$ are usually words, but as we see in Section 1.2.3, they need not be. This distribution is often decomposed into a product of conditional distributions on tokens. The most common approach is the so-called *causal* language model, which uses the following decomposition:

$$P(w_1, \dots, w_m) = \prod_{t=1}^{m} P(w_t \mid w_1, \dots, w_{t-1}). \tag{1.6}$$

Modeling the tokens one by one cannot only enable the model to factorize the handling of local information but also makes it easy to sample to generate new utterances. However most language models do not use an exact decomposition but either approximate $P(\boldsymbol{w})$ directly or use the decomposition of Equation 1.6 together with an approximation of the conditionals $P(w_t \mid w_1, \dots, w_{t-1})$. This is for example the case of word2vec which conditions each word on its close neighbors instead of using the whole sentence.

The use of language models is motivated by transfer learning, the idea that by solving a problem, we can get knowledge about a different but related problem. To assign a probability to a sequence, language models extract intermediate latent factors, which were proven to capture the semantic information contained in the sequence. Using these latent factors as distributed representations for natural language inputs improved the performance of most NLP tasks. The effectiveness of language models can be justified by the externalist approach and the distributional hypothesis exposed in Section 1.1: a word is defined by the distribution of the other words with which it co-occurs.

Since language models process sequences of words, we will delve into the details of these approaches in Section 1.3. Apart from the neural probabilistic language model of Bengio et al. (2003), a precursor to word embedding techniques was the CNN-based approach of Collobert and Weston (2008), both of them learn distributed word representations by approximating $P(\boldsymbol{w})$ using a window somewhat similar to word2vec.

All of these methods learn *static* word embeddings, meaning that the vector assigned to a word such as “bank” is the same regardless of the context in which the word appears. In the last few years, *contextualized* word embeddings have grown in popularity; in these approaches, the word “bank” is assigned different embeddings in the phrases “robbing a bank” and “bank of a river.” These methods were first based on recurrent neural networks (Section 1.3.2) such as ELMo but are now primarily based on transformers (Section 1.3.4). Among contextualized word embedding built using transformers, some are based on the causal decomposition of Equation 1.6 (e.g. GPT) while others are based on masked language models (e.g. BERT), a different approximation of $P(\boldsymbol{w})$ introduced in Section 1.3.4.2.

1.2.3 Subword Tokens

We defined word2vec and language models for a vocabulary V composed of words. This may seem natural in the case of English and other somewhat analytic languages,[18] but it cannot directly be applied to all languages. Furthermore, language models that work at the word level tend to have difficulties working with rare words. A first solution to this problem is to use character-level models, but these tend to have a hard time dealing with the resulting long sequences.

Modern approaches neither work at the word-level nor at the character-level; instead, an intermediate subword vocabulary is used. The standard

[18] An analytic language is a language with a low ratio of morphemes to words. This is in contrast to synthetic languages, where words have a complex inner structure. Take for example the Nahuatl word “Nimitztētlamaquiltīz” (I-you-someone-something-give-CAUSATIVE-FUTURE) meaning “I shall make somebody give something to you” (Suárez 1983). For this kind of language, word-level approaches fail. Older models preprocessed the text with a morphological segmentation algorithm, while modern approaches directly work on subword units.

method to build this vocabulary nowadays is to use the byte pair encoding algorithm (BPE, Gage 1994). BPE listed as Algorithm 1.1 consists in iteratively replacing the most common bigram c_1c_2 in a corpus with a new token c_{new}. This new token can then appear in the most common bigram with another token $c_{\text{new}}c_3$, they are then replaced with a new token c'_{new} which represents a tri-gram in the original alphabet: $c_1c_2c_3$. This is repeated until the desired vocabulary size is reached. In this way, BPE extracts tokens close to morphemes, the smallest linguistic unit with a meaning. As an example, by using this algorithm, the word "pretrained" can be split into three parts: "pre-," "-train-" and "-ed."

Word2vec can be both applied to words and to subwords extracted by BPE or other algorithms. This is the case of fastText (Bojanowski et al. 2017) which uses the word2vec algorithm on fixed-size subwords. All the models discussed in this section and the next have very loose requirements on the vocabulary V. However, they might work best using a smaller V; this is especially the case of transformers, the current state-of-the-art approach introduced in Section 1.3.4.

algorithm BPE
Inputs: n the vocabulary size
$\boldsymbol{t}$ the corpus
Output: V the vocabulary

$V \leftarrow$ all unique characters in $\boldsymbol{t}$
while $|V| < n$ **do**
  $c_1c_2 \leftarrow$ most common bigram in $\boldsymbol{t}$
  $c_{\text{new}} \leftarrow$ new token not in V
  $\boldsymbol{t} \leftarrow$ replace all occurrences of c_1c_2 in $\boldsymbol{t}$ by c_{new}
  $V \leftarrow V \cup \{c_{\text{new}}\}$
output V

Algorithm 1.1: The byte pair encoding algorithm.

1.3 Distributed Representation of Sentences

Most NLP tasks are tackled at the sentence level. In the previous section, we saw how to obtain representations of words. We now focus on how to aggregate these word representations in order to process whole sentences. Henceforth, given a sentence of length m, we assume symbolic words $\boldsymbol{w} \in V^m$ are embedded as $\boldsymbol{X} \in \mathbb{R}^{m\times d}$ in a vector space of dimension d. This can be achieved through the use of an embedding matrix $\boldsymbol{U} \in \mathbb{R}^{V\times d}$ such as the one provided by word2vec.

An early approach to sentence representation was to use *bag-of-words*, that is to simply ignore the ordering of the words. In this section, we focus on more modern, deep learning approaches. Section 1.3.1 presents CNNs, which process fixed-length sequences of words to produce representations of sentences. We then focus on RNNs in Section 1.3.2, a method to get representations of sentences through a causal language model. RNNs can be improved by an attention mechanism as explained in Section 1.3.3. Finally, we present transformers in Section 1.3.4, which build upon the concept of attention to extract state-of-the-art contextualized word representations.

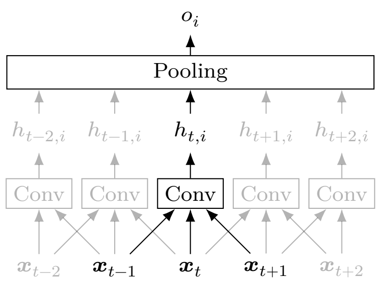


Figure 1.2: Architecture of a single convolutional filter with a pooling layer. The filter is of width 3, which means it works on trigrams. A single filter (the i-th) is shown here, this is repeated d' times, meaning that $\boldsymbol{h}_t, \boldsymbol{o} \in \mathbb{R}^{d'}$.

1.3.1 Convolutional Neural Network

Convolutional neural networks (CNN) can be used to build the representation of a sentence from the representation of its constituting words (Collobert and Weston 2008; Kim 2014). These words embeddings can come from word2vec (Section 1.2.1) or can be learned using a CNN with a language model objective (Section 1.2.2), the latter being the original approach proposed by Collobert and Weston (2008).

Collobert and Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning" ICML 2008

The basic idea behind CNN is to recognize patterns in a position-invariant fashion (Waibel et al. 1989). This is applicable to natural language following the principle of compositionality: the words composing an expression and the rules used to combine them determine its meaning, with little influence from the location of the expression in the text. So, given a sequence of d-dimensional embeddings $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathbb{R}^d$, a one dimensional CNN works on the n-grams of the sequence, that is the subwords[19] $\boldsymbol{x}_{t:t+n-1} = (\boldsymbol{x}_t, \dots, \boldsymbol{x}_{t+n-1})$ of length n. The basic design of a CNN is illustrated in Figure 1.2. A convolutional layer is parametrized by d' filters $\boldsymbol{W}^{(i)} \in \mathbb{R}^{n \times d}$ of width n and a bias $b^{(i)} \in \mathbb{R}$. The t-th output of the i-th filter layer is defined as:

[19] Here we use *subwords* in its formal language theory meaning. In the simple setting where we deal with words in a sentence, this *subword* actually designates a sequence of consecutive words.

$$h_t^{(i)} = f(\boldsymbol{W}^{(i)} * \boldsymbol{x}_{t:t+n-1} + b^{(i)}) \tag{1.7}$$

where $*$ is the convolution operator[20] and f is a non-linear function. As is usual with neural networks, several layers of this kind can be stacked. To obtain a fixed-size representation—which does not depend on the length of the sequence m—a pooling layer can be used. Most commonly, max-over-time pooling (Yamaguchi et al. 1990), which simply takes the maximum activation over time—that is sequence length—for each feature $i = 1, \dots, d'$.

[20] Usually, a cross-correlation operator is actually used, which is equivalent up to a mirroring of the filters when they are real-valued.

In the same way that word2vec produces a real vector space where words with similar meanings are close to each other, the sentence representations $\boldsymbol{o}$ extracted by a CNN tend to be close to each other when the sentences convey similar meanings. This is somewhat dependent on the task on which the CNN is trained. However, the purpose of CNN is usually to extract the semantics of a sentence, and the nature of most tasks makes it so that sentences with similar meanings should have similar representations.

1.3.2 Recurrent Neural Network

A limitation of CNNs is the difficulty they have modeling patterns of non-adjacent words. A second approach to process whole sentences is to use recurrent neural networks (RNN). RNNs purpose to sum up an entire sentence prefix into a fixed-size hidden state, updating this hidden state as the sentence is processed. This can be used to build a causal language model following the decomposition of Equation 1.6. As showcased by Figure 1.3, the hidden state $\boldsymbol{h}_t$ can be used to predict the next word w_{t+1} with a simple linear layer followed by a softmax, formally:

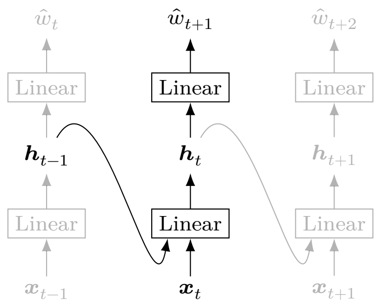


Figure 1.3: RNN language model unrolled through time.

$$\begin{aligned} \boldsymbol{h}_t &= f(\boldsymbol{W}^{(x)}\boldsymbol{x}_t + \boldsymbol{W}^{(h)}\boldsymbol{h}_{t-1} + \boldsymbol{b}^{(h)}) \\ \hat{w}_t &= \operatorname{softmax}(\boldsymbol{W}^{(o)}\boldsymbol{h}_t + \boldsymbol{b}^{(o)}) \end{aligned} \tag{1.8}$$

where $\boldsymbol{W}^{(x)}$, $\boldsymbol{W}^{(h)}$, $\boldsymbol{W}^{(o)}$, $\boldsymbol{b}^{(h)}$ and $\boldsymbol{b}^{(o)}$ are model parameters and f is a non-linearity, usually a sigmoid $f(x) = \sigma(x) = \frac{1}{1+\mathrm{e}^{-x}}$. This model is usually trained by minimizing the negative log-likelihood:

$$\mathcal{L}_{\text{RNN}}(\boldsymbol{\theta}) = \sum_{t=1}^{m} -\log P(w_t \mid \boldsymbol{x}_1, \dots, \boldsymbol{x}_{t-1}; \boldsymbol{\theta})$$

We generally use $\boldsymbol{\theta}$ to refer to the set of model parameters. In this case $\boldsymbol{\theta} = \{\boldsymbol{W}^{(x)}, \boldsymbol{W}^{(h)}, \boldsymbol{W}^{(o)}, \boldsymbol{b}^{(h)}, \boldsymbol{b}^{(o)}\}$.

using the backpropagation-through time algorithm. The gradient is run through all the steps of the RNN until reaching the beginning of the sequence. When the sequence is a sentence, this can easily be achieved. However, when longer spans of text are considered, the gradient only goes back a fixed number of tokens in order to limit memory usage.

1.3.2.1 Long Short-term Memory

Standard RNNs tend to have a hard time dealing with long sequences. This problem is linked to the vanishing and exploding gradient problems.

When the gradient goes through several non-linearities, it tends to be less meaningful, and gradient descent does not lead to satisfying parameters anymore. In particular, when $\boldsymbol{W}^{(h)}$ has a large spectral norm, the values $\boldsymbol{h}_t$ tend to get bigger and bigger with long sequences, on the other hand when its spectral norm is small, these values get smaller and smaller. When $\boldsymbol{h}_t$ has a large magnitude, the sigmoid activation saturates and $\frac{\partial \mathcal{L}_{\text{RNN}}}{\partial \boldsymbol{h}_t}$ gets close to zero, the gradient vanishes. RNN variants are used to alleviate this vanishing gradient problem, the most common being long short-term memory (LSTM, Hochreiter and Schmidhuber 1997).

Hochreiter and Schmidhuber, "Long Short-Term Memory" NECO 1997

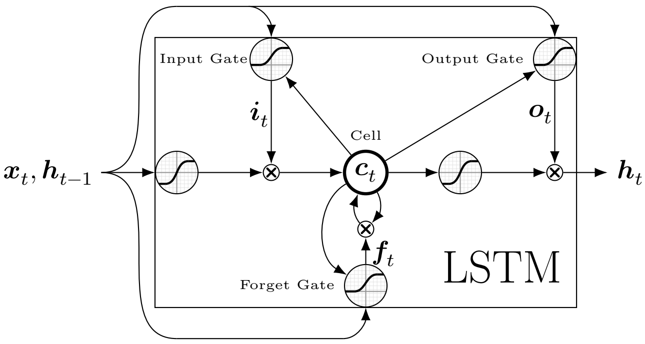


Figure 1.4: Architecture of an LSTM cell. In its simplest form, this block replaces the linear layer at the bottom of Figure 1.3. The link between $\boldsymbol{c}_t$ and $\boldsymbol{c}_{t-1}$ is illustrated by a self-loop but could be seen as an additional input and output.

LSTMs redefine the recurrence of RNNs (Equation 1.8) by adding multiplicative gates as illustrated by Figure 1.4. It is governed by the following set of equations:

$$
\begin{aligned}
\boldsymbol{x}'_t &= \begin{bmatrix} \boldsymbol{x}_t \\ \boldsymbol{h}_{t-1} \end{bmatrix} && \text{Recurrent input} \\
\tilde{\boldsymbol{c}}_t &= \tanh(\boldsymbol{W}^{(c)}\boldsymbol{x}'_t + \boldsymbol{b}^{(c)}) && \text{Cell candidate} \\
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}^{(i)}\boldsymbol{x}'_t + \boldsymbol{U}^{(i)}\boldsymbol{c}_{t-1} + \boldsymbol{b}^{(i)} && \text{Input gate} \\
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}^{(f)}\boldsymbol{x}'_t + \boldsymbol{U}^{(f)}\boldsymbol{c}_{t-1} + \boldsymbol{b}^{(f)}) && \text{Forget gate} \\
\boldsymbol{c}_t &= \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t + \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} && \text{New cell} \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}^{(o)}\boldsymbol{x}'_t + \boldsymbol{U}^{(o)}\boldsymbol{c}_t + \boldsymbol{b}^{(o)}) && \text{Output gate} \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) && \text{Hidden layer output}
\end{aligned}
$$

$\odot$ is the element-wise multiplication and σ the sigmoid function.

As with RNN, $\boldsymbol{\theta} = \{\boldsymbol{W}^{(c)}, \boldsymbol{W}^{(i)}, \boldsymbol{U}^{(i)}, \boldsymbol{W}^{(f)}, \boldsymbol{U}^{(f)}, \boldsymbol{W}^{(o)}, \boldsymbol{U}^{(o)}, \boldsymbol{b}^{(c)}, \boldsymbol{b}^{(f)}, \boldsymbol{b}^{(i)}, \boldsymbol{b}^{(o)}\}$ are model parameters.

The main peculiarity of LSTM is the presence of multiple gates used as masks or mixing factors in the unit. LSTM units are interpreted as having an internal cell memory $\boldsymbol{c}_t$ which is an additional (internal) state alongside $\boldsymbol{h}_t$ and is used as input of the cell alongside $\boldsymbol{x}_t$ and $\boldsymbol{h}_{t-1}$. When computing its activation, we first compute a cell candidate $\tilde{\boldsymbol{c}}_t$ which is the potential successor to $\boldsymbol{c}_t$. Then, the multiplicative gates come into play, the cell $\boldsymbol{c}_t$ is partially updated with a mix of $\boldsymbol{c}_{t-1}$ and $\tilde{\boldsymbol{c}}_t$ controlled by the input and forget gates $\boldsymbol{i}_t$ and $\boldsymbol{f}_t$. Finally, the output of the unit is masked by the output gate $\boldsymbol{o}_t$.[21]

[21] Note that the output gate $\boldsymbol{o}_t$ has its value computed from the new cell value $\boldsymbol{c}_t$ instead of $\boldsymbol{c}_{t-1}$ in contrast to the expression of $\boldsymbol{i}_t$ and $\boldsymbol{f}_t$.

It has been theorized (Hochreiter 1998) that the gates are what makes LSTMs so powerful. The multiplications allow the model to learn to control the flow of information in the unit, thus counteracting the vanishing gradient problem. The basic building block of multiplicative gates has been reused for other RNN cell designs such as gated recurrent unit (GRU, Cho et al. 2014). Furthermore, random cell designs using multiplicative gates can be shown to perform as well as LSTM (Greff et al. 2017). However, standard practice is to always use LSTM or GRU for recurrent neural networks.

1.3.2.2 ELMO

Recurrent neural networks with LSTM cells were widely used for language modeling, both at the character-level (Sutskever et al. 2011) and at the word-level (Jozefowicz et al. 2016). The first language model to become widely used for extracting contextual word embeddings was ELMo (Embeddings from Language Model, Peters et al. 2018) which uses several LSTM layers.

Peters et al., "Deep Contextualized Word Representations" NAACL 2018

The peculiarity of the word embeddings extracted by ELMo is that they are contextualized (see Section 1.2.2). Static word embeddings models like word2vec (Section 1.2.1) map each word to a unique vector. However, this fares poorly with polysemic words and homographs whose meaning depends on the context in which they are used. Contextualized word embeddings provide an answer to this problem. Given a sentence, ELMo proposes to use the hidden states $\boldsymbol{h}_t$ as a representation of each constituting word w_t. These representations are hence a function of the whole sentence.[22] Thus words are mapped to different vectors in different contexts.

Before ELMo, McCann et al. (2017) already trained contextualized word representations using an NMT task.

[22] In order to encode both the left and right context of a word, ELMo uses bidirectional LSTM, meaning that each layer contains two LSTM, one running from left-to-right and one right-to-left.

1.3.3 Attention Mechanism

To obtain a vector representation of a sentence from an RNN, two straightforward methods are to use the last hidden state $\boldsymbol{h}_m$ or use a pooling layer similar to the one used in CNN, such as max-over-time pooling. However, both of these approaches present shortcomings: the last hidden state tends to encode little information about the beginning of the sentence, while pooling is too indiscriminate and influenced by unimportant words. Using an attention mechanism is a way to avoid these shortcomings. Furthermore, an attention mechanism is parametrized by a *query* which allows us to select the piece of information we want to extract from the sentence.

The concept of attention first appeared in neural machine translation (NMT) under the name "alignment" (Bahdanau et al. 2015) before becoming ubiquitous in NLP. The same principle was also presented under the name *memory network* (Sukhbaatar et al. 2015; Weston et al. 2015). It is also the building block of transformers, which are presented next. With this in mind, we use the vocabulary of memory networks to describe the attention mechanism.

Bahdanau et al., "Neural Machine Translation by Jointly Learning to Align and Translate" ICLR 2015

1.3.3.1 Attention as a Mechanism for RNN

The principle of an attention layer on top of an RNN is illustrated by Figure 1.5. The layer takes three inputs: a query $\boldsymbol{q} \in \mathbb{R}^d$, memory keys $\boldsymbol{K} \in \mathbb{R}^{\ell\times d}$ and memory values $\boldsymbol{V} \in \mathbb{R}^{\ell\times d'}$. Originally, more often than not, $\boldsymbol{K} = \boldsymbol{V}$. In the model of Figure 1.5, the memory corresponds to the hidden states of the RNN, which was the most common architecture when attention was introduced in 2014. First, attention weights are computed from the query $\boldsymbol{q}$ and keys $\boldsymbol{K}$, then these weights are used to compute the output $\boldsymbol{o} \in \mathbb{R}^{d'}$ as a convex combination of the values $\boldsymbol{V}$:

$$\boldsymbol{o} = \operatorname{softmax}(\boldsymbol{K}\boldsymbol{q})\boldsymbol{V}. \tag{1.9}$$

Where softmax is a smooth version of the argmax function. It can also be seen as a multi-dimensional sigmoid, defined as:

$$\operatorname{softmax}(\boldsymbol{x})_i = \frac{\exp x_i}{\sum_j \exp x_j}$$

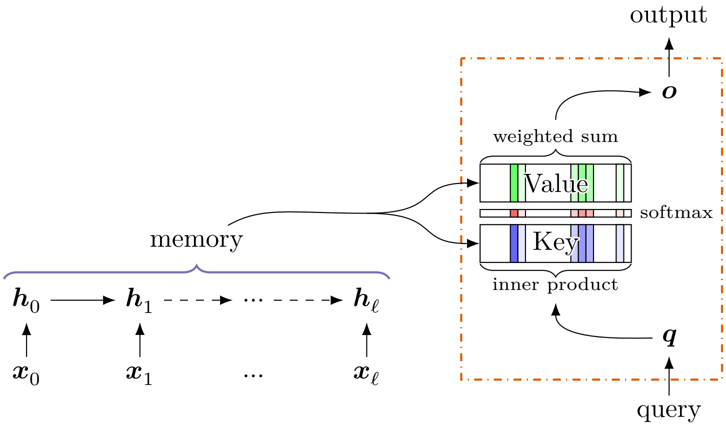


Figure 1.5: Schema of an attention mechanism. The attention scores are obtained by an inner product between the query and the memory. The output is obtained as a sum of the memory weighted by the softmax of the attention scores.

In NMT, the memory is built from the hidden states of an RNN running on the sentence to be translated (meaning $\ell = m$), while the query is the state of the translated sentence ("what was already translated"), the attention is then recomputed for each output position. In other words, a new representation of the source sentence is recomputed for each word in the target sentence. The attention weights—that is, the output of the softmax—can provide an interpretation of what the model is focusing on when making a prediction. In the case of NMT, the attention for producing a translated word usually focuses on the corresponding word or group of words in the source sentence.

1.3.3.2 Attention as a Standalone Model

Since the attention mechanism produces a fixed-size representation ($\boldsymbol{o}$) from a variable length sequence ($\boldsymbol{K}$, $\boldsymbol{V}$), it can actually be used by itself without an RNN. This was already mentioned in Sukhbaatar et al. (2015) and used for language modeling. We now succinctly present their approach. As shown Figure 1.6, this is a causal language model (Section 1.2.2), at each step $P(w_t \mid w_1, \dots, w_{t-1})$ is modeled. While the previous words constitute the memory of the attention mechanism, there is no natural value for the query. As such, for the first layer, it is simply taken to be a constant vector $q_i^{(1)} = 0.1$ for all $i = 1, \dots, d$. When several attention layers are stacked, the output $o^{(l)}$ of a layer l is used as the query $q^{(l+1)}$ for the layer $l + 1$. Furthermore, residual connections with linear layers and modified ReLU non-linearities[23] are introduced between layers thus: $\boldsymbol{q}^{(l+1)} = \mathrm{ReLU}_{◐}(\boldsymbol{W}^{(l)}\boldsymbol{q}^{(l)} + \boldsymbol{o}^{(l)})$ where the matrices $\boldsymbol{W}^{(l)} \in \mathbb{R}^{d\times d}$ are parameters of the model. As usual, the next word prediction $\hat{w}_i$ is made using a softmax layer.

Sukhbaatar et al., "End-To-End Memory Networks" NeurIPS 2015

[23] While the standard ReLU activation (Glorot et al. 2011) is defined as $\mathrm{ReLU}(x) = \max(0, x)$. The non-linearity used in this model is $\mathrm{ReLU}_{◐}$, which applies the ReLU activation to half of the units in the layer.

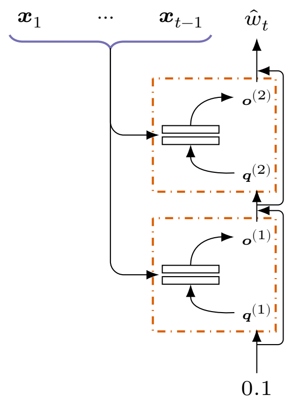


Figure 1.6: Schema of a memory network language model with two layers. Each red block corresponds to an attention mechanism as illustrated by Figure 1.5.

Temporal Encoding The attention mechanism as described above is invariant to a permutation of the memory. This is not a problem when an RNN is run on the sentence, as it can encode the relative positions of each token. However, in the RNN-less approach of Sukhbaatar et al. (2015) this information is lost, which is quite damaging for language modeling. Indeed, this would mean that shuffling the words in a sentence—like inverting the subject and object of a verb—does not change its meaning. In order to solve this problem, temporal encoding is introduced. When predicting

w_i, each word embedding $\boldsymbol{x}_j$ in the memory is summed with a relative position embedding $\boldsymbol{e}_{i-j}$. These position embeddings are trained through back-propagation like any other parameters.

Attention mechanisms form the basis of current state-of-the-art approaches in NLP. One of the explanations behind their success is that, in a sense, they are more shallow than RNN. Indeed, when computing $\frac{\partial \hat{w}_i}{\partial \boldsymbol{x}_j}$ for the language model of Sukhbaatar et al. (2015), one can see that part of the gradient goes through few non-linearities. In contrast, the information from $\boldsymbol{x}_j$ to $\hat{w}_i$ must go through the composition of at least $i-j$ non-linearities in an RNN, which may cause the gradient to vanish. However, an attention mechanism has linear complexity in the length of the sequence for a total of $\Theta(m \times d^2)$ operations at each step. When m is large, this can be prohibitive compared to RNN, which have a $\Theta(d^2)$ complexity at each step. On the other hand, an attention layer can easily be parallelized while an RNN always necessitates $\Omega(m)$ sequential operations.

1.3.4 Transformers

Transformers (Vaswani et al. 2017) were originally introduced for NMT. Likewise to the memory network language model presented above, they introduce several slight modifications of its architecture which make them the current state of the art for most NLP tasks. For conciseness, we present the concept of transformers as used by BERT (Bidirectional Encoder Representations from Transformers, Devlin et al. 2019). BERT is a language model used to extract contextualized embeddings similar to ELMo but using attention layers in place of LSTM layers.

Vaswani et al., "Attention is All you Need" NeurIPS 2017

Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" NAACL 2019

1.3.4.1 Transformer Attention

The attention layers used by transformers are slightly modified. First, it is often advisable that in a neural network, all activations follow a standard normal distribution $\mathcal{N}(0,1)$. In order to achieve this, transformers use scaled attention:

Note that in contrast to the classical attention mechanism presented in Section 1.3.3, transformers have $\boldsymbol{K} \neq \boldsymbol{V}$.

$$\operatorname{Attention}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \operatorname{softmax}\left(\frac{\boldsymbol{K}\boldsymbol{q}}{\sqrt{d}}\right)\boldsymbol{V}. \tag{1.10}$$

This ensures that if $\boldsymbol{K}$ and $\boldsymbol{q}$ follow a standard normal distribution, so does the input of the softmax.

Second, multi-head attention is used: each layer actually applies $h = 8$ attentions in parallel. To ensure each individual attention captures a different part of the semantic, its input is projected by different matrices, one for each attention head:

$$\operatorname{MultiHeadAttention}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \begin{bmatrix} \operatorname{head}_1(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) \\ \operatorname{head}_2(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) \\ \vdots \\ \operatorname{head}_h(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) \end{bmatrix} \boldsymbol{W}^{(o)}$$

$$\operatorname{head}_i(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \operatorname{Attention}(\boldsymbol{q}\boldsymbol{W}_i^{(q)}, \boldsymbol{K}\boldsymbol{W}_i^{(k)}, \boldsymbol{V}\boldsymbol{W}_i^{(v)}).$$

Lastly, on top of each attention layer is a linear layer with ReLU activation and a linear layer followed by layer normalization (Ba et al. 2016).

These linear layers are identical along the sequence length, akin to a convolution with kernel size 1. While the query of each layer is the output of the preceding layer, similarly to the model of Sukhbaatar et al. (2015), the initial query is now the current word itself $\boldsymbol{x}_t$. This architecture is illustrated in Figure 1.7.

Devlin et al. (2019) introduce two BERT architectures dubbed BERT-`small` and BERT-`large`. Like their names imply, BERT-`small` has fewer parameters than BERT-`large`, in particular, BERT-`small` is composed of 12 layers while BERT-`large` is composed of 24 layers.

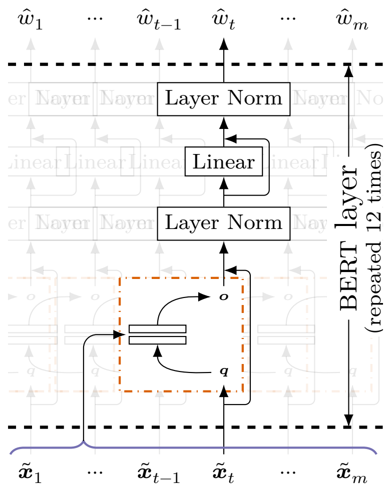


Figure 1.7: Schema of BERT, a transformer masked language model. The schema is focused on the prediction for a single position t, this is repeated for the whole sentence $t = 1, \dots, m$. The model presented is the BERT-`small` variant containing only 12 layers. The input vectors $\tilde{\boldsymbol{x}}_t$ are obtained from the corrupted sentence $\tilde{\boldsymbol{w}}$ using an embedding layer. To obtain $\hat{w}_t$ from the last BERT layer output, a linear layer with softmax over the vocabulary is used.

1.3.4.2 Masked Language Model

While some transformer models such as GPT (Generative Pre-Training, Radford et al. 2018) are causal language models, BERT is a *masked* language model (MLM). Instead of following Equation 1.6, the following approximation is used:

$$P(\boldsymbol{w}) \propto \prod_{t \in C} P(w_t \mid \tilde{\boldsymbol{w}}) \tag{1.11}$$

where C is a random set of indices, 15% of tokens being uniformly selected to be part of C, and $\tilde{\boldsymbol{w}}$ is a corrupted sequence defined as follow:

$$\tilde{w}_t = \begin{cases} w_t & & \text{if } t \notin C \\ \texttt{<}\text{BLANK}\texttt{/>} \text{ token} & \text{with probability } 80\% \\ \text{random token} & \text{with probability } 10\% & \text{if } i \in C \\ w_t & \text{with probability } 10\% \end{cases}$$

The masked tokens <BLANK/> make up the majority of the set C of tokens predicted by the model, thus the name “masked language model”. The main advantage of this approach compared to causal language model is that the probability distribution at a given position is parametrized by the whole sentence, including both the left and right context of a token.

1.3.4.3 Transfer Learning

The main purpose of BERT is to be used on a *downstream task*, transferring the knowledge gained on masked language modeling to a different problem. As with ELMo, the hidden state of the topmost layer, just before the linear and softmax, can be used as contextualized word representations. Furthermore, the first token, usually called “beginning of sentence” but dubbed CLS in BERT, can be used as a representation of the whole sentence.[24] In contrast with ELMo, BERT is usually fully fine-tuned on the downstream task. In the original article (Devlin et al. 2019), this was shown to outperform previous models on a wide variety of tasks, from question answering to textual entailments.

[24] This is by virtue of an additional *next sentence prediction* loss with which BERT is trained. We do not detail this task here as it is not essential to BERT’s training. Furthermore, the embedding of the CLS token is considered a poor representation of the sentence and is rarely used (Conneau and Lample 2019; Yang et al. 2019).

In this section, we presented several NLP models which allow us to get a distributed representation for words, sentences and words contextualized in sentences. These representations can then be used on a downstream task, such as relation extraction, as we do from Chapter 2 onward. We now focus on the other kind of data handled in this thesis: knowledge bases.

1.4 Knowledge Base

Our goal is to extract structured knowledge from text. In this section, we introduce the object we use to express this knowledge, namely the knowledge base. A knowledge base is a symbolic semantic representation of some piece of knowledge. It is defined by a set of concepts, named *entities*, and by the relationships linking these entities together, named *facts* or *statements*. Formally, a knowledge base is constructed from a set of entities $\mathcal{E}$, a set of relations $\mathcal{R}$ and a set of facts $\mathcal{D}_{\text{KB}} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. Note that these facts purpose to encode some kind of truth about the world. To illustrate, here are some examples from Wikidata (Vrandečić and Krötzsch 2014):

$$\begin{aligned}
\mathcal{E} &= \{\texttt{Q90}(\text{Paris}), \texttt{Q7251}(\text{Alan Turing}), \dots\} \\
\mathcal{R} &= \{\texttt{P1376}(\textit{capital of}), \texttt{P19}(\textit{place of birth}), \dots\} \\
\mathcal{D}_{\text{KB}} &= \{\texttt{Q90 P1376 Q142} \text{ (Paris is the capital of France)}, \\
&\qquad \texttt{Q3897 P1376 Q916} \text{ (Luanda is the capital of Angola)}, \\
&\qquad \texttt{Q7251 P19 Q122744} \text{ (Alan Turing was born in Maida Vale)}, \\
&\qquad \texttt{Q164047 P19 Q23311} \text{ (Alexander Pope was born in London)}, \\
&\qquad \dots\}
\end{aligned}$$

As indicated by the identifiers such as Q7251, knowledge bases link concepts together. An entity is a concept that may have several textual representations—surface forms—such as "Alan Turing" and "Alan Mathison Turing." Here, we showed the Wikidata identifier whose purpose is to identify concepts uniquely. For ease of reading, when there is no ambiguity between an entity and one of its surface forms, we simply write the surface form without the identifier of its associated concept.

Given two entities $e_1, e_2 \in \mathcal{E}$ and a relation $r \in \mathcal{R}$, we simply write $e_1\ r\ e_2$ as a shorthand notation for $(e_1, r, e_2) \in \mathcal{D}_{\text{KB}}$, meaning that r links e_1 and e_2 together. As illustrated by Figure 1.8, e_1 is called the *head entity* of the fact or *subject* of the relation r. Similarly, e_2 is called the *tail entity* or *object*, while r is called the *relation*, *property* or *predicate*.[25]

$$\underbrace{\overbrace{\text{Paris}^{\texttt{Q90}}}^{\text{head entity}}\ \overbrace{\textit{capital of}^{\texttt{P1376}}}^{\text{relation}}\ \overbrace{\text{France}^{\texttt{Q142}}}^{\text{tail entity}}}_{\text{fact}}$$

Figure 1.8: Structure of a knowledge base fact.

[25] The term *predicate* can either refer to the relation r, or to the couple (r, e_2), thus we will avoid using this terminology.

Thanks to this extremely rigid structure, knowledge bases are easier to process algorithmically. Querying some piece of information from a knowledge base is well defined and formalized. Query languages such as SPARQL ensure that information can be retrieved deterministically. This is in contrast to natural language, where querying some knowledge from a piece of text needs to be performed using an NLP model, thus incurring some form of variance on the result. With this in mind, it is not surprising that several machine learning models rely on knowledge bases to remove a source of uncertainty from their system; this can be done in a variety of tasks such as question answering (Berant et al. 2013; Yih et al. 2015), document retrieval (Dalton et al. 2014) and logical reasoning (Socher et al. 2013).

Example of SPARQL query for all capital cities in Asia:

```
SELECT ?capital
WHERE {
    ?capital capital of ?country.
    ?country part of Asia.
}
```

Commonly used general knowledge bases include Freebase (Bollacker et al. 2008), DBpedia (Auer et al. 2008) and Wikidata (Vrandečić and Krötzsch 2014). There are also several domain-specific knowledge bases such as Wordnet (G. A. Miller 1995) and GeneOntology (Gene Ontology Consortium 2004). Older works focus on Freebase—which is now discontinued—while newer ones focus on Wikidata and DBpedia. These knowledge bases usually include more information than what was described above. For example, Wikidata includes statement qualifiers that may modify a statement, such as the fact "Versailles capital of France" qualified by "end time: 5 October 1789." For the sake of simplicity, we limit ourselves to triplets in $\mathcal{E} \times \mathcal{R} \times \mathcal{E}$. Further details on the specific knowledge bases can be found in Appendix C.

1.4.1 Relation Algebra

Relations linking two entities from the same set of entities $\mathcal{E}$ are called binary endorelations. A relation such as "*capital of*" is a subset of the cartesian square $\mathcal{E}^2$; it is a set of pairs of entities linked together by this relation. The set of all possible such relations exhibit a structure called a relation algebra $(2^{\mathcal{E}^2}, \cap, \cup, ^-, \mathbf{0}, \mathbf{1}, \bullet, \boldsymbol{I}, \breve{})$. We use it as a formalized system of notation for relation properties. A relation algebra is defined from:

The concept of relation algebra was theorized as a structure for logical systems. Developed by several famous mathematicians such as Augustus De Morgan, Charles Peirce and Alfred Tarski, it can be used to express ZFC set theory. Here we only use relation algebra as a formal framework to express properties of binary relations.

- three special relations:
  - $\mathbf{0}$, the empty relation linking no entities together (e_1 $\mathbf{0}$ e_2 is always false);
  - $\mathbf{1}$, the complete relation linking all entities together (e_1 $\mathbf{1}$ e_2 is always true);
  - $\boldsymbol{I}$, the identity relation linking all entities to themselves (e_1 $\boldsymbol{I}$ e_2 is true if and only if $e_1 = e_2$).
- two unary operators:
  - the complementary relation $\bar{r}$ which links together entities not linked by r;
  - the converse $\breve{r}$ which reverses the direction of the relation such that e_1 $\breve{r}$ e_2 holds if and only if e_2 r e_1 holds.
- three binary operators (in order of lowest precedence, to highest precedence):
  - disjunction e_1 $(r_1 \cup r_2)$ e_2, either r_1 or r_2 link e_1 with e_2;
  - conjunction e_1 $(r_1 \cap r_2)$ e_2, both r_1 and r_2 link e_1 with e_2;
  - composition e_1 $(r_1 \bullet r_2)$ e_2, there exist $e_3 \in \mathcal{E}$ such that both e_1 r_1 e_3 and e_3 r_2 e_2 hold.

Thanks to this framework, we can express several properties on knowledge base relations since $\mathcal{R} \subseteq 2^{\mathcal{E}^2}$. For example, the *functional* property can be stated as $\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$. A relation r is functional when for all entities e_1 there is at most one entity e_2 such that e_1 r e_2 holds. The relation "*born in*" is functional since all entities are either born at a single place or not born at all. Taking the above definition this means that for all cities c if we take all entities who were born in c ($\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$) and then ($\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$) look at where these entities were born ($\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$), we must be back to c and only c ($\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$) or no such c shall exist ($\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$). We need to take the disjunction with $\boldsymbol{I}$ since some entities were not born anywhere, for example e $(\breve{r} \bullet r)$ e is false when r is "*born in*" and e is "Mount Everest."

Note that $\bullet$ composes relations in the opposite order of the function composition $\circ$. Indeed while $f \circ g$ means that g is applied first, then f is applied, "*mother* $\bullet$ *born in*" means that "*mother*" is first applied to the entity, then "*born in*" is applied to the result.

Other common properties of binary relations can be defined this way. One particular property of interest is the restriction of the domain and codomain of relations. A lot of relations can only apply to a specific type of entity, such as locations or people. To express these properties, we use the notation $\mathbf{1}_X \subseteq \mathbf{1}$ with $X \subseteq \mathcal{E}$ to refer to the complete relation restricted to entities in X: $\mathbf{1}_X = \{ (x_1, x_2) \mid x_1, x_2 \in X \}$. This allows us to define left-restriction (restriction of the domain) and right-restriction (restriction of the co-domain). Relevant properties are given in Table 1.1.

Property	Condition
Injective	$r \bullet \breve{r} \cup \boldsymbol{I} = \boldsymbol{I}$
Functional	$\breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$
Symmetric	$r = \breve{r}$
Transitive	$r \bullet r \cup r = r$
Left-restriction	$r \bullet \breve{r} \cup 1_X = 1_X$
Right-restriction	$\breve{r} \bullet r \cup 1_X = 1_X$

Table 1.1: Some fundamental relation properties expressed as conditions in relation algebra.

Some relation properties recurring in the literature are the cardinality constraints. They can be defined as combinations of the injective and functional properties:

Many-to-Many $(N \to N)$ the relation is neither injective nor functional. Examples: "author of," "language spoken," "sibling of."

Many-to-One $(N \to 1)$ the relation is functional but it is not injective. Examples: "place of birth," "country."

One-to-Many $(1 \to N)$ the relation is injective but it is not functional. Examples: "contains administrative territorial entity," "has part."

One-to-One $(1 \to 1)$ the relation is both injective and functional. Examples: "capital," "largest city," "highest point."

When a relation r is one-to-many, its converse $\breve{r}$ is many-to-one. The usual way to design relations in knowledge bases is to use many-to-one relations, making one-to-many relations quite rare in practice. Since most systems handle relations in a symmetric fashion, this has little to no effect.

Most of the examples given above are not strictly true. A person can be both registered as being born in Paris and in France. Some countries do not designate a single capital or share their highest point with a neighbor. However, defining these properties is helpful to evaluate the abilities of models to capture these kinds of relations. To handle such cases, these properties can be seen in a probabilistic way.[26]

[26] Given empirical data, the propensity of a relation to be many-to-one can be measured with a conditional entropy $\mathrm{H}(\mathrm{e}_2 \mid \mathrm{e}_1, r)$. An entropy close to zero means the relation tends to be many-to-one.

We use the notations from relation algebra to formalize assumptions made on the structure of knowledge bases. For example several models assume that $\forall r_1, r_2 \in \mathcal{R} : r_1 \cap r_2 = \mathbf{0}$, that is all pairs of entities are linked by at most one relation. A list of common assumptions is provided in Appendix B, it should prove useful from the Chapter 2 onwards. For readers unfamiliar with relation algebra notations, we provide detailed explanation of complex formulae in the margins throughout this thesis.

1.4.2 Distributed Representation through Knowledge Base Completion

One problem with knowledge bases is that they are usually incomplete. However, given some information about an entity, it is usually possible to infer additional facts about this entity. This is called *knowledge base completion.* Sometimes this inference is deterministic. For example, if two entities have the same two parents, we can infer that they are siblings. Quite often, this reasoning is probabilistic. For example, the head of state of a country usually lives in this country's capital; this probability can be further increased by facts indicating that previous heads of state died in the capital, etc.

The task of knowledge base completion is essential for our work because of two reasons. First of all, it is the standard approach to obtain a distributed representation of knowledge base objects. Second, the models used to tackle this task are often reused as part of relation extraction systems; this is the case of all approaches presented in this section.

We define two sub-tasks of knowledge base completion: *relation prediction* and *entity prediction.*[27] In the relation prediction task, the goal is to

[27] In the literature, both of these tasks can be called "link prediction" and "knowledge graph completion."

predict the relation between two entities (e_1 ? e_2), while entity prediction focuses on predicting a missing entity in a triplet (e_1 r ? or ? r e_2). Historically, this is performed using symbolic approaches. For example, this task can be tackled using an inference engine relying on a human expert inputting logical rules such as:

$$e_1 \textit{ parent of } e_2 \wedge e_1 \textit{ parent of } e_3 \wedge e_2 \neq e_3 \iff e_2 \textit{ sibling of } e_3,$$

or using the relation algebra notation introduced in Section 1.4.1:

$$\overparen{\textit{parent of}} \bullet \textit{parent of} \cap \bar{\boldsymbol{I}} = \textit{sibling of}.$$

Relation prediction is quite similar to our task of interest: relation extraction. The main difference being that relation prediction is defined on knowledge bases, while relation extraction takes natural language inputs. This parallel is exploited by the model presented in Chapter 3.

e_2 $\overparen{\textit{parent of}}$ e_1 means that e_1 is a parent of e_2. Adding a composition to this, e_2 $\overparen{\textit{parent of}} \bullet \textit{parent of}$ e_3 means that the aforementioned e_1 has a child e_3. This child e_3 could be the same as e_2, this is why we take the conjunction with the complement of the identity relation $\cap\bar{\boldsymbol{I}}$, thus obtaining the relation *sibling of*.

However, listing all possible logical implications is not feasible. As with NLP, to tackle this problem, another approach is to leverage distributed representations. Some good early results were obtained by RESCAL, which we present in Section 1.4.2.2. But the problem started to gather a lot of interest in the deep learning community with TransE (Section 1.4.2.3) which encodes relations as translation in the semantic space. This was followed by several other approaches that encoded relations as other kinds of geometric transformations. All the models presented in this section assume that the entities are embedded in a latent semantic space $\mathbb{R}^d$ with a matrix $\boldsymbol{U} \in \mathbb{R}^{\mathcal{E}\times d}$ where d is an hyperparameter.

1.4.2.1 Selectional Preferences

Selectional preferences is a simple formalism that purposes to encode each relation with two linear maps assessing the predisposition of an entity to appear as the head or tail of a relation in a true fact. This can be done using an energy formalism, where the energy of a fact is defined as:

$$\psi_{\text{SP}}(e_1, r, e_2) = \boldsymbol{u}_{e_1}^\top \boldsymbol{a}_r + \boldsymbol{u}_{e_2}^\top \boldsymbol{b}_r \tag{1.12}$$

with $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{\mathcal{R}\times d}$ two matrices encoding the preferences of each relation for certain entities. This energy function can then be used to define the probability that a fact holds using a softmax:

$$P(e_1, r, e_2) \propto \exp \psi_{\text{SP}}(e_1, r, e_2), \tag{1.13}$$

this is sufficient for entity and relation predictions as we can usually compute the partition function over the set of all entities or relations. If this is not feasible, a technique such as NCE (Section 1.2.1.2) or negative sampling (Section 1.2.1.3) can be used to approximate Equation 1.13. Still, selectional preferences do not encode the interaction of the head and tail entities. As such it is quite weak for entity prediction, thus more expressive models are needed.

1.4.2.2 RESCAL

RESCAL (Nickel et al. 2011) purposes to model relations by a bilinear form $\mathcal{E}\times\mathcal{E} \mapsto \mathbb{R}$ in the semantic space of entities. In other words, each relation $r \in \mathcal{R}$ is represented by a matrix $\boldsymbol{C}_r \in \mathbb{R}^{d\times d}$ with the training algorithm seeking to enforce the following property:

Nickel et al., "A Three-Way Model for Collective Learning on Multi-Relational Data" ICML 2011

$$\boldsymbol{u}_{e_1}^\top \boldsymbol{C}_r \boldsymbol{u}_{e_2} = \begin{cases} 1 & \text{if } e_1\ r\ e_2 \text{ holds} \\ 0 & \text{otherwise.} \end{cases}$$

This can be seen as trying to factorize the tensor of facts $\boldsymbol{X}$ as $\boldsymbol{U}\boldsymbol{C}\boldsymbol{U}^{\mathsf{T}}$, where $\boldsymbol{X} \in \{0,1\}^{\mathcal{E}\times\mathcal{R}\times\mathcal{E}}$ with $x_{e_1 r e_2} = 1$ if $e_1\ r\ e_2$ holds and $x_{e_1 r e_2} = 0$ otherwise. The parameters of the models $\boldsymbol{U}$ and $\boldsymbol{C}$ are trained using an alternating least-squares approach, minimizing a regularized reconstruction loss:

$$\mathcal{L}_{\text{RESCAL}}(\boldsymbol{X}; \boldsymbol{U}, \boldsymbol{C}) = \frac{1}{2} \sum_{\substack{e_1, e_2 \in \mathcal{E} \\ r \in \mathcal{R}}} (x_{e_1 r e_2} - \boldsymbol{u}_{e_1}^{\mathsf{T}} \boldsymbol{C}_r \boldsymbol{u}_{e_2})^2 + \frac{1}{2}\lambda(\|\boldsymbol{U}\|_F^2 + \sum_{r\in\mathcal{R}} \|\boldsymbol{X}_r\|_F^2) \tag{1.14}$$

Using bilinear forms allows RESCAL to capture entities interactions for each relation in a simple manner. However, the number of parameters to estimate grows quadratically with respect to the dimension of the semantic space d. This can be prohibitive as a large d is needed to ensure accurate modeling of the entities.

1.4.2.3 TransE

To find a balance between the number of parameters and the expressiveness of the model, geometric approaches were developed starting with TransE (Bordes et al. 2013). TransE proposes to leverage the regularity exhibited by Figure 1.1 to embed both entities and relations in the same vector space. Formally, its assumption is that relations can be represented as translations between entities' embeddings. In addition to representing each entity e by an embedding $\boldsymbol{u}_e \in \mathbb{R}^d$, each relation r is also embedded as a translation in the same space as $\boldsymbol{v}_r \in \mathbb{R}^d$. The idea being that if $e_1\ r\ e_2$ holds then $\boldsymbol{u}_{e_1} + \boldsymbol{v}_r \approx \boldsymbol{u}_{e_2}$. The authors argue that translations can represent hierarchical data by drawing a parallel with the embedding of a tree in an Euclidean plane—that is the usual representation of a tree as drawn on paper. As long as the distance between two levels in the tree is large enough, the children of a node are close together; this not only allows for the representation of one-to-many relations "child" but also for the many-to-many, symmetric and transitive relation "sibling" as the null translation.

Bordes et al., "Translating Embeddings for Modeling Multi-relational Data" NeurIPS 2013

In order to enforce the translation property, a margin-based loss is used to train an energy-based model. The energy of true triplets drawn from the knowledge base is minimized, while negative triplets are sampled and have their energy maximized up to a certain margin. Given a positive triplet (e_1, r, e_2) and a negative triplet (e'_1, r, e'_2), the TransE loss can be expressed as:

$$\mathcal{L}_{\text{TE}}(e_1, r, e_2, e'_1, e'_2) = \max\left(0, \gamma + \mathit{\Delta}(\boldsymbol{u}_{e_1} + \boldsymbol{v}_r, \boldsymbol{u}_{e_2}) - \mathit{\Delta}(\boldsymbol{u}_{e'_1} + \boldsymbol{v}_r, \boldsymbol{u}_{e'_2})\right), \tag{1.15}$$

where $\mathit{\Delta}$ is a distance function such as the squared Euclidean distance $\mathit{\Delta}(\boldsymbol{u}_{e_1} + \boldsymbol{v}_r, \boldsymbol{u}_{e_2}) = \|\boldsymbol{u}_{e_1} + \boldsymbol{v}_r - \boldsymbol{u}_{e_2}\|_2^2$. The negative triplets (e'_1, r, e'_2) are sampled by replacing one of the two entities of (e_1, r, e_2) by a random one which is sampled uniformly over all possible entities:

$$N(e_1, e_2) = \begin{cases} (e_1, e') & \text{with probability } 50\% \\ (e', e_2) & \text{with probability } 50\% \end{cases}$$
$$\text{with } e' \sim \mathcal{U}(\mathcal{E}).$$

Since d is a distance, when the loss $\mathcal{L}_{\text{TE}}$ is perfectly minimized, the positive part $+\mathit{\Delta}(\boldsymbol{u}_{e_1} + \boldsymbol{v}_r, \boldsymbol{u}_{e_2})$ is 0. This means that the negative part

$-\Delta(\boldsymbol{u}_{e'_1} + \boldsymbol{v}_r, \boldsymbol{u}_{e'_2})$ contributes to the loss only when it is smaller than the margin γ. Since this criterion depends on the distance between entities, it can easily be optimized by increasing the entity embeddings norms. To avoid this degenerate solution, the entity embeddings are renormalized at each training step. The training loop and initialization procedure are detailed in Algorithm 1.2. Parameters $\boldsymbol{U}$ and $\boldsymbol{V}$ are optimized by stochastic gradient descent with early-stopping based on validation performance.

algorithm TransE

Inputs: $\mathcal{D}_{\text{KB}}$ knowledge base
γ margin
d embedding dimension
b batch size

Outputs: $\boldsymbol{U}$ entity embeddings
$\boldsymbol{V}$ relation embeddings

▷ *Initialization* ◁

$\boldsymbol{U} \leftarrow \mathcal{U}_{|\mathcal{E}|\times d}\left(-\frac{6}{\sqrt{d}}, \frac{6}{\sqrt{d}}\right)$

$\boldsymbol{V} \leftarrow \mathcal{U}_{|\mathcal{R}|\times d}\left(-\frac{6}{\sqrt{d}}, \frac{6}{\sqrt{d}}\right)$

$\forall r \in \mathcal{R} : \boldsymbol{v}_r \leftarrow \boldsymbol{v}_r / \|\boldsymbol{v}_r\|_2$

▷ *Training* ◁

loop
  $\forall e \in \mathcal{E} : \boldsymbol{u}_e \leftarrow \boldsymbol{u}_e / \|\boldsymbol{u}_e\|_2$
  $B \leftarrow \emptyset$
  for $i = 1, \dots, b$ **do**
    Sample $(e_1, r, e_2) \sim \mathcal{U}(\mathcal{D}_{\text{KB}})$
    Sample $(e'_1, e'_2) \sim N(e_1, e_2)$
    $B \leftarrow B \cup \{(e_1, r, e_2, e'_1, e'_2)\}$
  Update $\boldsymbol{U}$ and $\boldsymbol{V}$ w.r.t.
  $$\nabla \sum_{(e_1,r,e_2,e'_1,e'_2)\in B} \mathcal{L}_{\text{TE}}(e_1, r, e_2, e'_1, e'_2)$$
output $\boldsymbol{U}, \boldsymbol{V}$

Algorithm 1.2: The TransE training algorithm. The relations are initialized randomly on the sphere but are free to drift away afterward, while entities are renormalized at each iteration. The loop updates parameters $\boldsymbol{U}$ and $\boldsymbol{V}$ using gradient descent and is stopped based on validation score. The gradient of $\mathcal{L}_{\text{TE}}$ is computed from Equation 1.15.

Evaluation The quality of the embeddings can be evaluated by measuring the accuracy of entity prediction based on them. Given a true triplet $(e_1, r, e_2) \in \mathcal{D}_{\text{KB}}$, the energy $\Delta(\boldsymbol{u}_{e'} + \boldsymbol{v}_r, \boldsymbol{u}_{e_2})$ is computed for all possible entities $e' \in \mathcal{E}$. The entity minimizing the energy is predicted as completing the triplet. The same procedure is then applied on e_2. The correct entity minimizes the energy quite rarely, therefore in order to have a more informative score Bordes et al. (2013) reports the mean rank of the correct entity among all the entities ranked by the energy of their associated triplets. For reference, on WordNet, the mean rank of the correct entity is 263 among 40 943 entities.

When expanding the expression $\Delta(\boldsymbol{u}_{e_1} + \boldsymbol{v}_r, \boldsymbol{u}_{e_2})$ where d is the Euclidean distance, the main term ends up being $\boldsymbol{u}_{e_1}^\top \boldsymbol{u}_{e_2} + \boldsymbol{v}_r^\top(\boldsymbol{u}_{e_2} - \boldsymbol{u}_{e_1})$. As such, TransE captures all 2-way interactions between e_1, r and e_2. However, this means that 3-way interactions are not captured, this is however standard in information extraction. Furthermore, TransE is unable to model several symmetric relations (when $r = \breve{r}$). To solve these problems, other geometric transformations were proposed to improve TransE expressiveness, such as first projecting entities on a hyperplane (TransH, Z. Wang et al. 2014) or having the entities and relations live in different spaces (TransR, Y. Lin et al. 2015). Finally, all the methods mentioned in this section are not only useful for entity and relation predictions, but also as methods to obtain distributed representations of knowledge bases entities and relations. The matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ learned by TransE can subsequently be used for other tasks involving knowledge bases, in the same way that transfer learning is used to obtain distributed representations of text using language models (Section 1.3.4.3).

1.5 Conclusion

As exposed in Section 1.1, we are in the middle of a transition away from symbolic representations towards distributed ones. We inscribe this thesis within this transition. We deal with two kinds of symbolic representations of meaning: unstructured language and structured knowledge bases. In this chapter, we presented methods to extract distributed representations for both of these systems. While in the following chapters, we will deal with the link between language and knowledge bases.

Following word2vec (Section 1.2.1), feature extraction for textual inputs is now mostly done through word embeddings. In order to obtain a representation of a sentence, the models on top of these word embeddings progressively evolved from CNN (Section 1.3.1) and RNN (Section 1.3.2) towards transformers and contextualized word embeddings (Section 1.3.4). As we will see in the next chapter, this trend was exactly followed by relation extraction models.

We then introduce the structured knowledge representation we handle throughout this thesis, knowledge bases. In particular, Section 1.4.1 gives a formal notation for handling relations which we use to write modeling hypotheses in subsequent chapters. Finally, Section 1.4.2 presents common models making use of distributed representations of knowledge bases for the task of knowledge base completion. This task is not only the usual evaluation framework for distributed knowledge base representations but is also of special interest for Chapter 3, where we leverage the similarity between the knowledge base completion and the relation extraction tasks.

The progression of models presented in this chapter also reflects a progression of the scale of problems. We started by exploring the representation of words, one of the smallest semantic units, then moved on to sentences, then to knowledge bases, which purpose to represent whole pans of human knowledge. Another underlying thread to this chapter is the notion of relationship. While the idea is quite pervasive in Section 1.4, it is also present in Section 1.2 through the not-so-randomly chosen example of Figure 1.1.[28] Even in Section 1.3, representations of sentences are obtained by modeling the relationship of words with each other. For example, in a transformer, the attention weights capture the relationship between two words: the query and one element of the memory.

[28] This figure presented the word embeddings of some countries and their capitals. The relationship between the words seems to bear the same regularity as the relationship between the underlying entities. This regularity being representative of the *capital of* relationship.

In the next chapter, we make the link between the two symbolic representations of meaning we studied: language and knowledge bases. More specifically, we present relation extraction models. State-of-the-art models build heavily on the distributed representations methods introduced in this chapter and are the main focus of this thesis.

Chapter 2

Relation Extraction

> *When two objects, qualities, classes, or attributes, viewed together by the mind, are seen under some connexion, that connexion is called a relation.*
>
> — Augustus De Morgan, "On the Syllogism, No. III, and on Logic in general" (1864, p. 203)

> *Hard constraints are the midwife to good design.*
>
> — Maciej Cegłowski, *Web Design: The First 100 Years* (2014)

The rapid increase in the amount of published information brings forward the problem of how to handle large amounts of data. To this goal, *information extraction* aims at discovering the underlying semantic structure of texts. As such, it is considered to be a part of natural language understanding. It is the link from unstructured text to structured data. Following Section 1.4, we will use knowledge bases as a formalization of structured data. However, to encompass the notion of information more appropriately, the concept of knowledge base needs to be taken in a broad sense. The strict definition of knowledge underlying most knowledge bases only includes general facts and does not encompass things such as "Seneca is contemptuous even of the best garum." However, this sentence conveys a piece of information that needs to be considered by information extraction systems. As such, we will consider text-specific facts such as "Seneca *dislikes* garum" to be facts belonging in a knowledge base.

In this thesis, we focus on relation extraction, a subtask of information extraction. Precursors of relation extraction were the template filling tasks. In these tasks, objects corresponding to a given class—usually a specific kind of event—must be extracted from a text, and a template must be filled with information about this object. This was pioneered by Sager (1972) but started gathering interest with the message understanding conferences (MUC) supported by DARPA.[29] The template filling task was formalized and evaluated in a systematic way starting with MUC-2[30] in 1989. But it was not until 1997 that MUC-7 formalized the modern relation extraction task. The MUCs were succeeded by the automatic content extraction (ACE) program convened by the NIST[31] starting in 1999.

In contrast to relation extraction, when filling a template about an entity, the template has a fixed number of fields to be filled, in the language of Section 1.4.1, this means that all relations are left-total: $r \bullet \breve{r} = r \bullet \breve{r} \cup \boldsymbol{I}$.

Sager, "Syntactic Formatting of Science Information" AFIPS 1972

[29] The Defense Advanced Research Projects Agency, a research agency of the USA Department of Defense.

[30] At the time, the conference was known as MUCK-II.

[31] The National Institute of Standards and Technology, an agency of the USA Department of Commerce.

The main information extraction task is known as *knowledge base population* and consists in generating knowledge base facts from a set of documents. This task can be broken down into several steps, as illustrated by Figure 2.1:

Entity chunking seeks to locate entities in text. A similar task is named entity recognition (NER) which not only locates the entities but also assigns them with a type such as "organization," "person," "location," etc. The relation extraction datasets we consider in subsequent chapters do not include this entity-type information. However, NER was more prevalent in relation extraction works during the 2000s decade.

Entity linking assigns a knowledge base entity identifier to a tagged entity in a sentence. This disambiguates "Paris, France" Q90, from "Paris, son of Priam, king of Troy" Q167646 and "Paris, genus of the true lover's knot plant" Q162121. Following the above discussion on our broad sense of knowledge, an entity may not necessarily appear in an existing knowledge base, in which case the entity identifier can be taken to be the entity's surface form.

Relation extraction assigns a knowledge base relation identifier to an ordered pair of tagged entities in a sentence. Paris is not only the capital of France, it is also located in France. However, the sentence of Figure 2.1 does not convey the idea of location but the one of capital, thus predicting "*located in country*" P17 would be incorrect there.

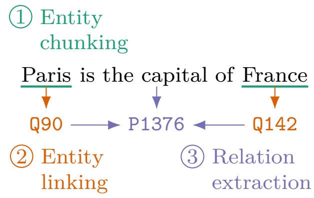


Figure 2.1: The three standard tasks for knowledge base population. First, entity chunking locates the entities in the sentence, here "Paris" and "France." Second, entity linking map each entity to a knowledge base identifier, here Q90 and Q142. Third, relation extraction find the relation linking the two entities, here P1376 (*capital of*).

Whereas Chapter 1 introduces the main tools used in relation extraction systems, the present chapter focuses on the relation extraction task itself. We formally define relation extraction in Section 2.1 and introduce its main variants encountered in the literature. A fundamental problem of relation extraction models is how to obtain supervision. Hand labeling a dataset is tedious and error-prone, so several alternative supervision techniques have been considered over the years; this is the focus of Section 2.2. We then introduce noteworthy supervised approaches–including weakly and semi-supervised ones—in Sections 2.3 and 2.4. As we will see in Section 2.1, the task can be tackled at the sentence level or at a higher level. Section 2.3 introduces sentence-level models, while Section 2.4 introduces higher-level models. Lastly, we delve into the main subject of this thesis, unsupervised relation extraction, in Section 2.5. Each of these sections is generally ordered following historical development, with older methods appearing first and current state-of-the-art appearing last.

2.1 Task Definitions

The relation extraction task was shaped by several datasets with different goals. The first MUCs focused on detecting naval sightings and engagement in military messages. Subsequent conferences moved towards the extraction of business-related relations in news reports. Nowadays, general encyclopedic knowledge is usually extracted from either news reports or encyclopedia pages. Another common goal is to extract drugs, chemical and symptoms interactions in biomedical texts (Lee et al. 2019). For further details, Appendix C contains a list of datasets with information about the source of the text and the nature of the relations to be extracted. Depending on the end-goal for which relation extraction is used, different definitions of the task might be more fitting. We now formally define the relation extraction task and explore its popular variants.

In relation extraction, we assume that information can be represented as a knowledge base $\mathcal{D}_{\text{KB}} \subseteq \mathcal{E}^2 \times \mathcal{R}$ as defined in Section 1.4. In addition to the set of entities $\mathcal{E}$ and the set of relations $\mathcal{R}$, we need to define the source of information from which to extract relations. The information source can come in several different forms, but we use a single basic definition on sentences which we can refine later on. We assume entity chunking was performed on our input data. We only deal with binary relations[32] since they are the ones commonly encoded in knowledge bases. We can therefore

For ease of notation, we changed the placement of entities in the tuple corresponding to a fact from the one used in Section 1.4. This will allow us to refer to the entity pair as $\boldsymbol{e} \in \mathcal{E}^2$.

[32] As described in Section 1.4.1, this means that only relations between two entities are considered. Moreover, higher-arity relations can be decomposed into sets of binary ones.

define $\mathcal{S}$ as a set of sentences with two tagged and ordered entities:

$$\begin{aligned}\mathcal{S} = \{&\text{“}\underline{\text{Jan Kasl}}_{e_1}\text{ became mayor of }\underline{\text{Prague}}_{e_2}\text{.”},\\ &\text{“}\underline{\text{Vincent Callebaut}}_{e_2}\text{ was born in 1977 in }\underline{\text{Belgium}}_{e_1}\text{.”},\\ &\dots\}.\end{aligned}$$

In this example, two sentences are given; in each sentence, the relation we seek is the one between the two entities marked by underlines. The entities need to be ordered since most relations are asymmetric ($r \neq \breve{r}$). In practice, this means that one entity is tagged as e_1 and the other as e_2. The standard setting is to work on sentences; this can of course be generalized to larger chunks of text if needed.

Relation extraction can also be performed on semi-structured documents, such as a Wikipedia page with its infobox or an HTML page that might contain lists and tables. This is the case of DIPRE presented in Section 2.3.2. As long as the semi-structured data can be represented as a token list, and standard text models can still be applied.

The tagged entities inside the sentences of $\mathcal{S}$ are not the same as entities in knowledge bases. They are merely surface forms. These surface forms are not sensu stricto elements of $\mathcal{E}$. Indeed, the same entity can have several different surface forms, and the same surface form can be linked to several different entities depending on context. To map these tagged surface forms to $\mathcal{E}$, entity linking is usually performed on the corpus. In practice, this means that we consider samples from $\mathcal{S} \times \mathcal{E} \times \mathcal{E}$. Finally, since the two tagged entities are ordered, we simply assume that the first entity in the tuple corresponds to the entity tagged e_1 in the sentence, while the second entity refers to e_2.[33] If entity linking is not performed on the dataset, we can simply assume that the surface forms are actually entities, in this case, and in this case alone, $\mathcal{E}$ is a set of surface forms. This is somewhat uncommon, the standard practice being to have linked entities.

[33] Note that e_2 can appears before e_1 in the sentence.

Also, note that this setup is still valid for sentences with three or more entities, as we can consider all possible entity pairs:

$$\begin{aligned}\mathcal{S} = \{&\text{“}\underline{\text{Alonzo Church}}_{e_1}\text{ was born on June 14, 1903, in }\underline{\text{Washington, D.C.}}_{e_2}\text{, where his father, Samuel Robbins Church, was the judge of the Municipal Court for the District of Columbia.”},\\ &\text{“}\underline{\text{Alonzo Church}}_{e_2}\text{ was born on June 14, 1903, in Washington, D.C., where his father, }\underline{\text{Samuel Robbins Church}}_{e_1}\text{, was the judge of the Municipal Court for the District of Columbia.”},\\ &\dots\}.\end{aligned}$$

In this example, we give two elements from $\mathcal{S}$, these elements are different since their markings $\underline{}_e$ differ. We often use the word sentence without qualifications to refer to elements from $\mathcal{S}$. Still, even though the two sentences above are the same in the familiar sense of the term, they are different in our definition.

Now, given a sentence with two tagged, ordered, and linked entities, we can state the goal of relation extraction as finding the semantic relation linking the two entities as conveyed by the sentence. Since the set of possible relations is designated by $\mathcal{R}$, we can sum up the relation extraction task as finding a mapping taking the form:

$$\boxed{f_{\text{sentential}}\colon \mathcal{S} \times \mathcal{E}^2 \to \mathcal{R}} \tag{2.1}$$

When we have access to a supervised dataset, all the information (head entity, relation, tail entity, conveying sentence) is provided. Table 2.1 gives some supervised samples examples. We denote a dataset of sentences with tagged, ordered, and linked entities as $\mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}^2$ and a supervised dataset

Head	Relation	Tail	Sentence
Q210175 MI5	P159 headquarters location	Q198519 Thames House	The exterior and interior of Freemasons' Hall continued to be a stand-in for Thames House$_{e_2}$, the headquarters of MI5$_{e_1}$.
Q210175 MI5	P101 field of work	Q501700 counter-intelligence	Golitsyn's claims about Wilson were believed in particular by the senior MI5$_{e_1}$ counterintelligence$_{e_2}$ officer Peter Wright. Wright, Peter (1987)
Q158363 SMERSH	P101 field of work	Q501700 counter-intelligence	In its counter-espionage$_{e_2}$ and counter-intelligence roles, SMERSH$_{e_1}$ appears to have been extremely successful throughout World War II.
Q198519 Thames House	P466 occupant	Q210175 MI5	The Freemasons' Hall in London served as the filming location for Thames House$_{e_1}$, the headquarters for MI5$_{e_2}$.

Table 2.1: Samples from the FewRel dataset. The surface forms in the head, relation and tail columns are only given for ease of reading and are usually not provided.

as $\mathcal{D}_\mathcal{R} \subseteq \mathcal{D} \times \mathcal{R}$. Given an entity pair $\boldsymbol{e} = (e_1, e_2)$, a sample in which these entities appear (s, e_1, e_2) is called a *mention*. A sample which convey a fact $e_1\ r\ e_2$ is called an *instance* of r.

Mentions as defined here can be called "entity mentions," while instances may be referred to as "relation mentions."

The relation extraction task as stated by Equation 2.1 is called *sentential extraction*. It is the traditional relation extraction setup, the sentences are considered one by one, and a relation is predicted for each sentence separately. However, information can be leveraged from the regularities of the dataset itself. Indeed, some facts can be repeated in multiple sentences, in which case a model could enforce some kind of consistency on its predictions. Even beyond a simple consistency of the relations predicted, in the same fashion that a word can be defined by its context, so can an entity. This kind of regularities can be exploited by modeling a dependency between samples even when conditioned on the model parameters. While tackling relation extraction at the sentence level might be sufficient for some datasets, others might benefit from larger context, especially when the end goal is to build a knowledge base containing general facts. This gives rise to the *aggregate extraction* setting, in which a set of tagged sentences is directly mapped to a set of facts without a direct correspondence between individual sentences and individual facts.

$$\boxed{f_\text{aggregate}\colon 2^{\mathcal{S}\times\mathcal{E}^2} \to 2^{\mathcal{E}^2\times\mathcal{R}}} \tag{2.2}$$

The left-hand side of Equation 2.2 is a subset of $\mathcal{S} \times \mathcal{E}^2$, that is $\mathcal{D}$ or a subset thereof. On the right-hand side, we have a subset of $\mathcal{E}^2 \times \mathcal{R}$; we tintend to find $\mathcal{D}_\text{KB}$ or a subset thereof. However, each individual sample $(s, \boldsymbol{e}) \in \mathcal{D}$ does not need to be mapped to an individual fact $(\boldsymbol{e}, r) \in \mathcal{D}_\text{KB}$.

Quite often in this case, the problem is tackled at the level of entity pairs, meaning that instead of making a prediction from a sample in $\mathcal{S} \times \mathcal{E}^2$, the prediction is made from $2^\mathcal{S} \times \mathcal{E}^2$. This setup is required for multi-instance approaches presented in Section 2.4.2. Aggregate extraction may impose a relatively more transductive approach[34] since predictions rely directly on previously observed samples. Usually, aggregate models still extract some form of prediction at the sentence level, even if they do not need to. Therefore, the key point of aggregate approaches is the explicit handling of dataset-level information. Some models may heavily depend on this global information, to the point that they cannot be trained without some form of repetition in the dataset. The sentential–aggregate distinction constitutes a spectrum. While all unsupervised methods exhibit some aggregate traits, they do not necessarily exploit as much structural information as they could; this is the key point of Chapter 4.

[34] Transductive approaches are contrasted to inductive approaches. In the inductive approach—such as neural networks—parameters $\boldsymbol{\theta}$ are estimated from the training set. When labeling on an unknown sample, the model makes its prediction only from parameters $\boldsymbol{\theta}$ and the unlabeled sample, access to the training set is no longer necessary. This is called induction since "rules" ($\boldsymbol{\theta}$) are obtained from examples. On the other hand,

2.1.1 Nature of Relations

in the transductive approach–such as K-NN—observations on the train set are directly transferred to test samples without first generalizing to a set of rules.

The supervised relation extraction task described above is quite generic. The approaches to tackle it in practice vary quite a lot depending on the specific nature of the facts we seek to extract and the corpus structure. In this subsection, we present some variations on the nature of $\mathcal{R}$ commonly encountered in the literature.

2.1.1.1 Unspecified Relation: *Other*

The set $\mathcal{R}$ is built using a finite set of labels. These labels do not describe the relationship between all entities in all possible sentences. Indeed some entities are deemed unrelated in some sentences. A distinction is sometimes made between relation extraction and relation detection, depending on whether a relation is assumed to exist between the two entities in a sentence or not. This apparent absence of relation is often called "*other*," since a relation between the two entities might exist but is simply not present in the relation schema considered (Hendrickx et al. 2010). In this case, we can still use the usual relation extraction setup by augmenting $\mathcal{R}$ with the following relation:

Hendrickx et al., "SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals" SemEval 2010

$$\textit{other} = \bigcap_{r\in\mathcal{R}} \bar{r}. \tag{2.3}$$

We use the notation of Section 1.4.1 where $\bar{r}$ refers to the complementary relation of the named relations r in the schema $\mathcal{R}$. Note that using the definition of relations as a set of entity pairs is not strictly correct here since two entities may be linked by a relation that is simply not conveyed by a specific sentence containing them. The underlying problem to this notational conundrum is the fact that *other* is only needed for mono-relation extraction when one and exactly one relation must be predicted for a sample; see Section 2.4.2 for an alternative. The definition given in Equation 2.3 is nonetheless fitting to the widespread distant supervision setting which we describe Section 2.2.2.

However note that "*other*" is not a relation like the others, it is defined by what it is not instead of being defined by what it is. This peculiarity calls for special care on how it is handled, especially during evaluation.

2.1.1.2 Closed-domain Assumption

As stated above, the set $\mathcal{R}$ is usually built from a finite set of labels such as *parent of* and *part of*. This is referred to as the *closed-domain assumption*. Another approach is to consider $\mathcal{R}$ is not known beforehand (Banko et al. 2007). In particular open information extraction (OIE, Section 2.5.2) directly uses surface forms as relation labels. In this case, the elements of $\mathcal{R}$ are strings of words, not defined in advance, and even potentially not-finite. We can see OIE as a preliminary task to relation extraction: the set of surface forms can be mapped to a traditional closed-set of labels. When $\mathcal{R}$ is not known beforehand, the relation extraction problem can be called *open-domain relation discovery*. This is the usual setup for unsupervised relation extraction described in Section 2.5.

Banko et al., "Open Information Extraction from the Web" IJCAI 2007

2.1.1.3 Directionality and Ontology

Most relations r are not symmetric ($r \neq \breve{r}$). There are several different approaches to handle this asymmetry. In the SemEval 2010 Task 8 dataset (Section C.6), the first entity in the sentence is always tagged e_1, and the second is always tagged e_2. The relation set $\mathcal{R}$ is closed under the converse operation (Hendrickx et al. 2010):

$$\forall r\in\mathcal{R} : \breve{r}\in\mathcal{R}.$$

This is the most common setup. In this case, the relation labels incorporate the directionality; for example, the SemEval dataset contains both *cause–effect*(e_1, e_2) and *cause–effect*(e_2, e_1) depending on whether the first entity appearing in the sentence is the cause or the effect. This means that given a $r \in \mathcal{R}$ in the SemEval dataset, we can easily query the corresponding $\breve{r}$. On the other hand, the relation set of the FewRel dataset (Section C.2) is not closed under the converse operation (Han et al. 2018). Furthermore, it is a mono-relation dataset without *other*. This means that all samples $(s, e_1, e_2) \in \mathcal{D}$ convey a relation between e_1 and e_2. Naturally, in this case, the entity tagged e_2 may appear before the one tagged e_1. And indeed, for relations that do not have their converse in $\mathcal{R}$, the same sentence s with the tags reversed may not appear in the FewRel dataset since this would need to be categorized as $\breve{r} \notin \mathcal{R}$.

Han et al., "FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation" EMNLP 2018

In general, the order of e_1 and e_2 is not fixed. This is particularly true in the open-domain relation setup, when $\mathcal{R}$ being unknown, can not be equipped with the converse operation. In this case, it is common to feed the samples in both arrangements: with the first entity tagged e_1 and the second e_2, and the reverse: with the first entity tagged e_2 and the second e_1. This can be seen as a basic data augmentation technique.

More generally, the relation set $\mathcal{R}$ might possess a structure called a *relation ontology*. This is especially true when $\mathcal{R}$ comes from a knowledge base such as Wikidata (Vrandečić and Krötzsch 2014). In this case, $\mathcal{R}$ can be equipped with several operations other than the converse one. For example, Wikidata endows $\mathcal{R}$ with a subset operation, the relation *parent organization* `P749` is recorded as a subset of *part of* `P361`, such that e_1 *parent organization* $e_2 \implies e_1$ *part of* e_2, or using the notation of Section 1.4.1: *parent organization* $\cup$ *part of* $=$ *part of*.

2.1.2 Nature of Entities

The approach to tackle the relation extraction task also quite heavily depends on the nature of entities. In particular, an important distinction must be made on whether the *unique referent assumption* is postulated. This has been the case in most examples given thus far. For instance, "Alan Turing" designates a single human being, even if several people share this name; we only designate one of them with the entity `Q7251` "Alan Turing." However, this is not always the case, for example, in the following sample from the SemEval 2010 Task 8 dataset:

> The $\underline{\text{key}}_{e_1}$ was in a $\underline{\text{chest}}_{e_2}$.
> Relation: *content–container*(e_1, e_2)

SemEval 2010 Task 8 is one of those datasets without entity linking, which is rather common when dealing with non-unique referents.

In this case, the entities "key" and "chest" do not always refer to the same object. The relation holds in the small world described by this sentence, but it does not always hold for every object designated by "key". This is closely related to the fineness of entity linking. Indeed, one could link the surface form "key" above with an entity designating this specific key, but this is not always the case, as exemplified by the SemEval 2010 Task 8 dataset. This distinction is pertinent to the relation extraction task, especially in the aggregate setting. When applied to entities with a unique referent, the *content–container*(e_1, e_2) relation is $N \to 1$ or at least transitive. However, when the unique referent assumption is false, this relation is not $N \to 1$ anymore since several "key" entities can refer to different objects located in different containers.

The aggregate setup is not necessarily contradictory with the unique referent assumption. Even though not all "keys" are in a "chest," this fact still gives us some information about "keys," in particular they can be in a "chest," which is not the case of all entities.

The unique referent assumption is not binary; the distinction is quite fuzzy in most cases. Should the entity `Q142` "France" refers both to the modern country and to the twelfth-century kingdom? What about the

West Frankish Kingdom? How should we draw the distinction? Instead of categorizing the model on whether they take the unique referent assumption for granted, we should instead look at their capacity to capture the kind of relationship between a key and a chest as conveyed by the above sample.

More generally, all the usual properties of grammatical nouns can lead to variations of the relation extraction task. For example, many models focus on rigid designators such as "Lucius Junius Brutus" which are opposed to flaccid designators such as "founder of the Roman Republic." Both refer to the same person Q223440. However, it is possible to imagine a world where the "founder of the Roman Republic" does not refer to Q223440. On the contrary, if Q223440 exists, "Lucius Junius Brutus" ought to refer to him.

Finally, another variation of the definition of entities commonly encountered in relation extraction comes from coreference resolution. Some datasets resolve pronouns such that in the sentence "$\underline{\text{She}}_e$ died in Marylebone," the word "she" can be considered an entity linked to Q7259 "Ada Lovelace" if the context in which the sentence appears supports this. In this case, the surface form of the entity gives little information about the nature of the entity. This can be problematic for models relying too heavily on entities' surface forms. In particular, early relation extraction models did not have access to entity identifiers; at the time, pronoun entities were avoided altogether.

2.2 The Problem of Data Scarcity

Ideally, a labeled dataset should be available for the source language and target relation domain $\mathcal{R}$, but alas, this is rarely the case. In particular, the order of $\mathcal{R}$ can range in the thousands, in which case, accurate labeling is tedious for human operators. To circumvent this problem, alternative supervision strategies have been used.

Despite the ubiquity of the terms, it is not easy to define the different forms of supervision clearly. We use the following practical definition: a dataset is supervised if among its features, one—the labels—must be predicted from the others. Furthermore, to distinguish with the self-supervised setup, we need to impose that the labels must be somewhat hard to obtain, typically through manual annotation.[35] For our task at hand, a supervised dataset takes the form $\mathcal{D}_\mathcal{R} \subseteq \mathcal{S} \times \mathcal{E}^2 \times \mathcal{R}$, indeed we seek to predict relation labels and obtaining those is tedious and error-prone. On the other hand, an unsupervised dataset takes the form $\mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}^2$, which is much easier to obtain: vast amounts of text are now digitized and can be processed by an entity chunker and an entity linker. An intermediate supervision setting is semi-supervision when a small subset of samples are supervised while other are left unsupervised, which can be stated as $\mathcal{D}_\text{semi} \subseteq \mathcal{S} \times \mathcal{E}^2 \times (\mathcal{R} \cup \{\varepsilon\})$.[36]

Despite these different kinds of datasets on which a relation extraction model can be trained, evaluating such a model is nearly always done using a supervised dataset $\mathcal{D}_\mathcal{R}$. In this section, we present two other approaches to train a model without manual labeling: bootstrap and distant supervision.

[35] To add to the confusion, the distinction between self-supervised and unsupervised is not necessarily pertinent, e.g. Yann LeCun retired "unsupervised" from his vocabulary, replacing it with "self-supervised" (LeCun and Misra 2021). In this case, the difficulty of obtaining the labels might be the sole difference between the "unsupervised/self-supervised" and "supervised" setups.

[36] Here, we denote by ε the absence of labels for a sample since this is often reflected by an empty field.

2.2.1 Bootstrap

Another method to deal with the scarcity of data is to use bootstrap. Early approaches to relation extraction often focused on a single relation and fell into this category of bootstrapped methods. The bootstrap process (Algorithm 2.1) starts with a small amount of labeled data and finds extraction rules by generalizing to a large amount of unlabeled data. As such, it is a semi-supervised approach. We now describe this algorithm by following the work that pioneered this approach.

algorithm BOOTSTRAP
  Inputs: $\mathcal{D}$ unlabeled dataset
    O or R seed
  Outputs: O occurrences
    R rules
  Start with either O or R
  loop
    $O \leftarrow \{x \in \mathcal{D} \mid R \text{ matches on } x\}$
    $R \leftarrow$ induce rules from occurrences O
  output O, R

Algorithm 2.1: The bootstrap algorithm. Occurrences are simply a set of samples $O \subseteq \mathcal{D}$ conveying the target relation. The algorithm can be either seeded with a set of occurrences O (Brin 1999) or a set of rules R (Hearst 1992). When starting with a set of occurrences, the algorithm must first start by extracting a set of rules, then alternate between finding occurrences and rules as listed.

Hearst (1992) propose a method to detect a single relation between noun phrases: hyponymy. They define e_1 to be an hyponym of e_2 when the sentence "An e_1 is a (kind of) e_2." is acceptable to an English speaker. This relation is then detected inside a corpora using lexico-syntactic patterns such as:[37]

> e_1 ,? including (e_2,)* (or|and)? e_3
> $\implies$ e_2 *hyponym of* e_1
> $\implies$ e_3 *hyponym of* e_1

[37] The syntax used here is inspired by regular expression: "()" are used for grouping, "?" indicates the previous atom is optional, "|" is used for alternatives and "*" is the Kleene star meaning zero or more repetitions.

where the entities e_i are constrained to be noun phrases. This rule matches on the following sentence:

> All common-law countries, including Canada and England…
> $\implies$ Canada *hyponym of* Common-law country
> $\implies$ England *hyponym of* Common-law country

Hearst (1992) proposes the following process: start with known facts such as hyponym(England, Country), find all places where the two entities co-occur in the corpus and write new rules from the patterns observed, which allows them to discover new facts to repeat the process with. Beside some basic lemmatization—which explains why "countries" became "country" in the example above—all noun phrases are treated as possible entities. This is sensible since the end goal of the approach is to generate new facts for the WordNet knowledge base. In Hearst (1992), writing new rules was not done automatically but performed manually.

Hearst, "Automatic Acquisition of Hyponyms from Large Text Corpora" COLING 1992

Following equation 2.1, a sentential relation extraction system usually defines a relation r as a subset of $\mathcal{S} \times \mathcal{E} \times \mathcal{E}$, i.e. relations are conveyed jointly by sentences and entity pairs. In contrast, Hearst (1992) makes the following assumption:

Assumption $\mathscr{H}_{\text{PULLBACK}}$: *It is possible to find the relation conveyed by a sample by looking at the entities alone and ignoring the sentence; and conversely by looking at the sentence alone and ignoring the entities.* $\mathcal{D} = \mathcal{S} \times_{\mathcal{R}} \mathcal{E}^2$.

The assumption of Hearst (1992) is that there are two morphisms $\mathcal{S} \to \mathcal{R}$ and $\mathcal{E}^2 \to \mathcal{R}$, therefore $\mathcal{D}$ must have a form which makes this decomposition possible: $(s, \boldsymbol{e}) \in \mathcal{D}$ if and only if s and $\boldsymbol{e}$ are mapped to the same relation. In other words, $\mathcal{D}$ completes the two relation extraction morphisms to a commutative square:

$$\begin{tikzcd} \mathcal{D} \arrow[r] \arrow[d] & \mathcal{S} \arrow[d] \\ \mathcal{E}^2 \arrow[r] & \mathcal{R} \end{tikzcd}$$

In category theory, this object is called a pullback and noted $\times_{\mathcal{R}}$. This also means that given a sample from $\mathcal{D}$, it is possible to find its relation without looking at its sentence or its entities since either of them is sufficient.

This implies that given a pair of entities, whatever is the sentence in which they appear, the conveyed relation is the same. On the contrary, given a sentence, the conveyed relation is always the same, whatever the entities. As such the representation of a relation is split into two parts:

a set of entity pairs $r_{\mathcal{E}} \subseteq \mathcal{E}^2$, which can be represented exactly;

a set of sentences $r_{\mathcal{S}} \subseteq \mathcal{S}$, which in Hearst (1992) was represented by a set of patterns matching only sentences in $r_{\mathcal{S}}$, such as "e_1 ,? including (e_2,)* (or|and)? e_3."

Given a dataset $\mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}^2$, it is possible to map from $r_{\mathcal{E}}$ to $r_{\mathcal{S}}$ by taking all sentences where the two entities appear and vice-versa by taking all pairs of entities appearing in the given sentences. The second process $\mathcal{R}_{\mathcal{S}} \times \mathcal{D} \to \mathcal{R}_{\mathcal{E}}$ is straightforward to implement exhaustively. While the first process $\mathcal{R}_{\mathcal{E}} \times \mathcal{D} \to \mathcal{R}_{\mathcal{S}}$ was performed manually by Hearst (1992).

2.2.2 Distant Supervision

Craven and Kumlien, "Constructing biological knowledge bases by extracting information from text sources" ISMB 1999

Craven and Kumlien (1999) introduced the idea of weak supervision to relation extraction as a compromise between hand labeled dataset and unsupervised training. It was then popularized by Mintz et al. (2009) under the name *distant supervision*. Their idea is to use a knowledge base $\mathcal{D}_{\text{KB}} \subseteq \mathcal{E}^2 \times \mathcal{R}$ to supervise an unsupervised dataset $\mathcal{D}$. The underlying assumption can be stated as:

Mintz et al., "Distant supervision for relation extraction without labeled data" ACL 2009

The use of assumptions or modeling hypotheses noted $\mathscr{H}_{\text{NAME}}$ is central to several relation extraction models, especially unsupervised ones. We strongly encourage the reader to look at the list of assumptions in Appendix B. The appendix provides counter-examples when appropriate. Furthermore, it lists the sections in which each assumption was introduced for reference.

Assumption $\mathscr{H}_{\text{DISTANT}}$: *A sentence conveys all the possible relations between all the entities it contains.*
$\mathcal{D}_{\mathcal{R}} = \mathcal{D} \bowtie \mathcal{D}_{\text{KB}}$
where $\bowtie$ denotes the natural join operator:

$$\mathcal{D} \bowtie \mathcal{D}_{\text{KB}} = \{\, (s, e_1, e_2, r) \mid (s, e_1, e_2) \in \mathcal{D} \wedge (e_1, e_2, r) \in \mathcal{D}_{\text{KB}} \,\}.$$

In other words, each sentence $(s, e_1, e_2) \in \mathcal{D}$ is labeled by all relations r present between e_1 and e_2 in the knowledge base $\mathcal{D}_{\text{KB}}$. This is sometimes referred to as an unaligned dataset, since sentences are not aligned with their corresponding facts. The assumption $\mathscr{H}_{\text{DISTANT}}$ is quite obviously false, and is only used to build a supervised dataset. A classifier is then trained on this dataset. In most works, including the one of Mintz et al. (2009), the model is designed to handle the vast amount of false positive in $\mathcal{D} \bowtie \mathcal{D}_{\text{KB}}$, usually through the aggregate extraction setting (see Section 2.1).

A caveat of distantly supervised datasets is that evaluation is often complex. Mintz et al. (2009) evaluate their approach on Freebase (Section C.3) by holding-out part of the knowledge base. However, the number of false negatives forces them to manually label the facts as true or false themselves.

2.3 Supervised Sentential Extraction Models

In the supervised setup, all variables listed in Table 2.1 are given at train time. During evaluation, the relation must be predicted from the other three variables: sentence, head entity and tail entity. The predictions for each sample can then be compared to the gold standard.[38] We introduce the commonly used metric for evaluation on a supervised dataset in Section 2.3.1. The following sections focus on important supervised methods, including weakly-supervised and semi-supervised methods. These sections focus on sentential relation extraction methods, which realize Equation 2.1. In contrast, Section 2.4 focuses on aggregate methods, which often build upon sentential approaches.

[38] When a distant supervision dataset is used, "gold standard" is somewhat a misnomer. In this case, the relation labels are often referred to as a "silver standard" since they are not as good as possible.

2.3.1 Evaluation

Since supervised relation extraction is a standard multiclass classification task, it uses the usual F_1 metric, with one small tweak to handle directionality. As for training, we use samples from $\mathcal{D}_{\mathcal{R}} \subseteq \mathcal{S} \times \mathcal{E}^2 \times \mathcal{R}$ for evaluation. Let's call $x \in \mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}^2$ an unlabeled sample, and $g\colon \mathcal{D} \to \mathcal{R}$ the function which associates with each sample x its gold label in the dataset (as given by $\mathcal{D}_{\mathcal{R}}$). Similarly, let's call $c\colon \mathcal{D} \to \mathcal{R}$ the function which associates with each sample x the relation predicted by the model we are evaluating. The standard F_1 score for a relation $r \in \mathcal{R}$ can be defined as:

$$\text{precision}(g, c, r) = \frac{|\{\, x \in \mathcal{D} \mid c(x) = g(x) = r \,\}|}{|\{\, x \in \mathcal{D} \mid c(x) = r \,\}|} = \frac{\text{true positive}}{\text{predicted positive}}$$

$$\begin{aligned}
\operatorname{recall}(g, c, r) &= \frac{|\{\, x \in \mathcal{D} \mid c(x) = g(x) = r \,\}|}{|\{\, x \in \mathcal{D} \mid g(x) = r \,\}|} = \frac{\text{true positive}}{\text{labeled positive}} \\
F_1(g, c, r) &= \frac{2}{\operatorname{precision}(g, c, r)^{-1} \times \operatorname{recall}(g, c, r)^{-1}}.
\end{aligned}$$

To aggregate these scores into a single number, multiple approaches are possible. First of all, micro-averaging: the true positives, predicted positive and labeled positive are averaged over all relations. In the case where all samples have one and only one label and prediction, micro-precision, micro-recall and micro-F_1 collapse into the same value, namely the accuracy. However, when computing a micro-metric on a dataset containing the *other* relation (Section 2.1.1.1), the samples labeled *other* are ignored, making the difference between micro-precision and micro-recall relevant again.

The second set of approaches uses macro-averaging, which means that the scores are averaged a first time for each relation before taking the average of these averages over the set of relations. This compensates for the class imbalance in the dataset since when taking the average of the averages, the score for a rare class is weighted the same as the score for a frequent class. The "directed" macro-scores are defined as usual:

$$\begin{aligned}
\overrightarrow{\operatorname{precision}}(g, c) &= \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \operatorname{precision}(g, c, r) \\
\overrightarrow{\operatorname{recall}}(g, c) &= \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \operatorname{recall}(g, c, r) \\
\overrightarrow{F_1}(g, c) &= \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} F_1(g, c, r).
\end{aligned}$$

However, two other variants exist. These variants try to discard the orientation of the relationship by packing together a relation r with its reverse $\breve{r}$. This allows us to evaluate separately the ability of the model to find the correct relation and to find which entity is the subject (e_1) and which is the object (e_2). The simplest way to achieve this is to simply ignore the orientation:

$$\overleftrightarrow{\operatorname{precision}}(g, c) = \frac{1}{|\mathcal{R}^\dagger|} \sum_{\{r, \breve{r}\} \in \mathcal{R}^\dagger} \frac{\big|\{\, x \in \mathcal{D} \mid c(x), g(x) \in \{r, \breve{r}\} \,\}\big|}{\big|\{\, x \in \mathcal{D} \mid c(x) \in \{r, \breve{r}\} \,\}\big|},$$

where $\mathcal{R}^\dagger$ is the set of relations paired by ignoring directionality. The set $\mathcal{R}^\dagger$ is well defined, since for the datasets using this metric, $\mathcal{R}$ is closed under the reverse operation $\breve{}$ with the notable exception of *other*. However, similarly to micro-metrics, *other* is often ignored altogether. It only influences the final metrics through the degradation of recall on samples mispredicted as *other* and of precision on samples mispredicted as not *other*. Following the definitions above, we can similarly define $\overleftrightarrow{\operatorname{recall}}$ and $\overleftrightarrow{F_1}$.

Finally, as a compromise between the directed $\overrightarrow{F_1}$ and undirected $\overleftrightarrow{F_1}$, the half-directed metric was designed:

$$\overleftharpoon{\operatorname{precision}}(g, c) = \frac{1}{|\mathcal{R}^\dagger|} \sum_{\{r, \breve{r}\} \in \mathcal{R}^\dagger} \frac{\big|\{\, x \in \mathcal{D} \mid g(x) \in \{r, \breve{r}\} \land c(x) = g(x) \,\}\big|}{\big|\{\, x \in \mathcal{D} \mid c(x) \in \{r, \breve{r}\} \,\}\big|}.$$

The key difference with the undirected metric is that while the prediction and gold must still be equal to r or $\breve{r}$, they furthermore need to be equal to each other. Figure 2.2 gives a visual explanation using the confusion matrix. Note that the distinction between directed and undirected metrics can also apply to micro-metrics.

In conclusion, the evaluation of supervised approaches varies along three axes:

- Whether *other* is considered a normal relation or is only taken into account through degraded precision and recall on the other classes.
- Whether the directionality of relations is taken into account.
- Whether class imbalance is corrected through macro-aggregation.

We now describe supervised relation extraction models, starting in this section with sentential approaches.

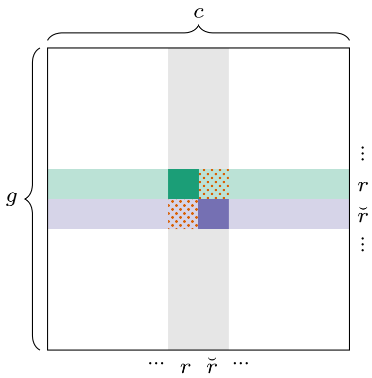


Figure 2.2: Supervised metrics defined on the confusion matrix. Directed metrics consider green and blue to be different classes, the $\overrightarrow{\text{recall}}$ for the relation r is computed by dividing the number of samples in the dark green cell by the total number of samples in the green row. Undirected metrics consider green and blue to be the same class, the $\overleftrightarrow{\text{recall}}$ for this class is computed by summing the four cells in the center including the two hatched ones and dividing by the sum of the two rows. Half-directed metrics also consider $\{r, \breve{r}\}$ to form a single class but the $\overleftarrow{\text{recall}}$ is computed by summing the two dark cells in the center—ignoring the two hatched ones—and dividing by the sum of the two rows.

2.3.2 Regular Expressions: DIPRE

Dual Iterative Pattern Relation Expansion (DIPRE, Brin 1999) follows the bootstrap approaches (Section 2.2.1) and thus assumes $\mathscr{H}_{\text{PULLBACK}}$. Compared to Hearst (1992), DIPRE proposes a simple automation for the $\mathcal{R}_\mathcal{E} \times \mathcal{D} \to \mathcal{R}_\mathcal{S}$ step—the extraction of new patterns—and applies it to the extraction of the "*author of book*" relation. To facilitate this automation and in contrast to Hearst (1992), it limits itself to two entities per patterns. DIPRE introduces the split-in-three-affixes technique illustrated by Figure 2.3. The entities split the text into three parts: prefix before the first entity, infix between the two entities and suffix after the second entity. This could be considered five parts with the two entities' surface forms since they are not part of any of the three affixes. This split reappeared in other works since, with the simplest methods assuming that the infix alone conveys the relation. Even in the case of DIPRE, all three affixes are considered, but the infix needed to match exactly, while the prefix and suffix could be shortened in order to make a pattern more general. All patterns are specific to an URL prefix, which made the algorithm pick up quickly on lists of books, with the algorithm also handling patterns where the author appeared before the title with a simple boolean marker.

Brin, "Extracting Patterns and Relations from the World Wide Web" WebDB 1999

$$\underbrace{\texttt{<li>}}_{\text{prefix}}\ \overbrace{\text{TITLE}}^{e_1}\ \underbrace{\texttt{}\ \text{by}}_{\text{infix}}\ \overbrace{\text{AUTHOR}}^{e_2}\ \underbrace{\texttt{(}}_{\text{suffix}}$$

Figure 2.3: DIPRE split-in-three-affixes method. The algorithm ran on HTML code, `<li>` marks a list item, while `<b></b>` surrounds bold text.

In order to generate new patterns, DIPRE takes all occurrences with the same infix and with the title and author in the same order. To avoid pattern which are too general they use the following approximation of the specificity of a pattern:

$$\begin{aligned}\text{specificity}(\text{pattern}) &= -\log(P(\text{pattern matches}))\\ &\approx \text{total length of the affixes.}\end{aligned}$$

When this specificity is lower than a given threshold divided by the number of known books it matched, the pattern was rejected. In the experiment, the algorithm was run on a starting set of five (author, title) facts which generated three patterns, one of which is given in Figure 2.3; these patterns produced in turn 4 047 facts. As per Hearst (1992), the algorithm was then iterated once again on these new facts. The second iteration introduced bogus facts, which were removed manually. Finally, the third iteration produced a total of 15 257 *author of book* facts. Brin (1999) manually analyzes twenty books out of these 15 257 and found that only one of them was not a book but an article, while four of them were obscure enough not to appear in the list of a major bookseller.

A limitation of the bootstrap approaches assuming $\mathscr{H}_{\text{PULLBACK}}$ is that this assumption naively entails the following:

As a reminder from Section 1.4.1: $\mathbf{0}$ denotes the empty relation linking no entities together. So $r_1 \cap r_2 = \mathbf{0}$ should be understood as "if we take the relation linking together all the entity pairs connected at the same time ($\cap$) by r_1 and r_2, we should obtain the relation liking no entities together ($\mathbf{0}$)."

Assumption $\mathscr{H}_{\text{1-ADJACENCY}}$: *There is no more than one relation linking any two entities.*

$\forall r_1, r_2 \in \mathcal{R}: r_1 \cap r_2 = \mathbf{0}$

Indeed, if a pair of entities is linked by two relations, this would implies a sentence containing these two entities also convey the two relations. By induction it follows that the two relations would actually be the same.

The approach of DIPRE was subsequently used by other systems such as Snowball (Agichtein and Gravano 2000), which uses more complex matching and pattern generation algorithms and formalizes the experimental setup. We now focus on another semi-supervised approach similar to bootstrap, which was important to the development of relation extraction methods.

2.3.3 Dependency Trees: DIRT

D. Lin and Pantel, "DIRT – Discovery of Inference Rules from Text" KDD 2001

Discovery of Inference Rules from Text (DIRT, D. Lin and Pantel 2001) also uses the $\mathscr{H}_{\text{PULLBACK}}$ assumption but makes a single iteration of the bootstrap algorithm from a single example. Furthermore, DIRT makes the pattern building $\mathcal{R}_\mathcal{E} \times \mathcal{D} \to \mathcal{R}_\mathcal{S}$ more resilient to noise and applies the algorithm to multiple relations. Another difference is that it factorizes the definition of $\mathcal{R}_\mathcal{S}$ using dependency paths instead of regular expressions. Given a sentence, a dependency parser can create a tree where nodes are built from words, and the arcs between the nodes correspond to the grammatical relationship between the words. This is called a dependency tree and is exemplified by Figure 2.4. After building a dependency tree, we can take the path between two nodes in the tree, for example the path between "John" and "problem" in the tree of Figure 2.4 is:

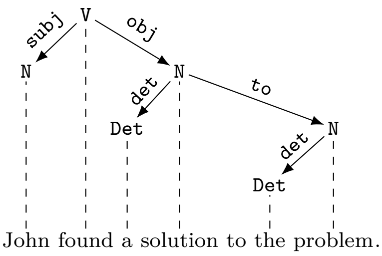


Figure 2.4: Example of dependency tree given by D. Lin and Pantel (2001) generated using the Minipar dependency parser. The nodes correspond to words in the sentence, as indicated by the dashed line. Each node is tagged by the part-of-speech (POS) of the associated word. The arrows between the nodes are labeled with the dependency between the words. The following abbreviations are used: `N` is noun, `V` is verb, `Det` is determiner, `subj` is subject, `obj` is object, and `det` is the determiner relation.

$$\leftarrow\texttt{N:subj:V}\leftarrow\text{find}\rightarrow\texttt{V:obj:N}\rightarrow\text{solution}\rightarrow\texttt{N:to:N}\rightarrow$$

Note that lemmatization is performed on the nodes. D. Lin and Pantel (2001) state their assumption as an extension of the distributional hypothesis (see section 1.1):

Distributional Hypothesis on Dependency Paths: *If two dependency paths occur in similar contexts, they tend to convey similar meanings.*

> *While hunting in Africa, I shot an elephant in my pajamas. How he got into my pajamas, I don't know.*
> — Groucho Marx, Animal Crackers (1930)
>
> The ambiguity of the prepositional phrase "in my pajamas" would be removed by a dependency tree. It can either be linked to the noun "elephant" or to the verb "shot."

In the case of DIRT, context is defined as the two endpoints of the paths. For example, the context of the path given above in Figure 2.4 consists of the words "John" and "problem." As such, this can be seen as a probabilistic version of the $\mathcal{R}_\mathcal{E} \times \mathcal{D} \to \mathcal{R}_\mathcal{S}$ step. In order to ensure these paths correspond to meaningful relations, only paths between nouns are considered. For example, by counting all entities appearing at the endpoints of the path above, D. Lin and Pantel (2001) observe that the following path have similar endpoints:

$$\leftarrow\texttt{N:subj:V}\leftarrow\text{solve}\rightarrow\texttt{V:obj:N}\rightarrow$$

Therefore, they can conclude that these two paths correspond to the same relation. The orientation of a path is not essential. If the subject of "solve" appears after its object in a sentence, we still want this path to be counted the same as the one above. As introduced in Section 2.1.1.3, this is a common problem in relation extraction. To solve this in a relatively straightforward manner, we simply assume all paths come in the two possible orientations, so for each sentence, the extracted path and its reverse are added to the dataset. We use a mutual information-based measure to evaluate how similar two set of endpoints are. Since counting all possible pairs would be too memory intensive—the squared size of the vocabulary $|V|^2$ is usually in the order of the billion or more—we measure the similarity of the first and second endpoint separately. To measure the preference of the dependency path π to have the word $w \in V$ appears at the endpoint $\ell \in \{\leftarrow, \rightarrow\}$, the following conditional pointwise mutual information is used:

The similarity metric equations in D. Lin and Pantel (2001) are quite informal. In particular, they do not state that ℓ has a special role as a conditional variable in the pmi and erroneously designate the same value as $\operatorname{mi}(\pi, m, \ell)$. The equations given here are our own.

$$\begin{aligned}\operatorname{pmi}(\pi, w \mid \ell) &= \log \frac{P(\pi, w \mid \ell)}{P(\pi \mid \ell)P(w \mid \ell)} \\ &= \log \frac{P(\pi, \ell, w)P(\ell)}{P(\pi, \ell)P(\ell, w)}.\end{aligned}$$

This quantity can be computed empirically using a hash table counting how many time the triplet (π, ℓ, w) appeared in the dataset. We can then compute the similarity between two paths given an endpoint ℓ then take the geometric average for the two possible value of ℓ to obtain an unconditioned similarity between paths:

$$\operatorname{sim}(\pi_1, \pi_2, \ell) = \frac{\sum_{w \in C(\pi_1, \ell) \cap C(\pi_2, \ell)} \big(\operatorname{pmi}(\pi_1, w \mid \ell) + \operatorname{pmi}(\pi_2, w \mid \ell)\big)}{\sum_{w \in C(\pi_1, \ell)} \operatorname{pmi}(\pi_1, w \mid \ell) + \sum_{w \in C(\pi_2, \ell)} \operatorname{pmi}(\pi_2, w \mid \ell)}$$

$$\operatorname{sim}(\pi_1, \pi_2) = \sqrt{\operatorname{sim}(\pi_1, \pi_2, \leftarrow) \times \operatorname{sim}(\pi_1, \pi_2, \rightarrow)},$$

where $C(\pi, \ell)$ designates the context, that is the set of words appearing at the endpoint ℓ of the path π.

Using this similarity function, D. Lin and Pantel (2001) can find sets of paths corresponding to particular relations by looking at frequent paths above a fixed similarity threshold. They evaluate their method manually on a question answering dataset. For each question, they extract the corresponding path and then look at the 40 most similar paths in their dataset and manually tag whether these paths would answer the original question. The accuracy of DIRT ranges from 92.5% for the relation "*manufactures*" to 0% for the relation "*monetary value of*" for which no similar paths were found.

2.3.4 Hand-designed Feature Extractors

The first supervised systems for relation extraction were designed for the template relations (TR) task of the seventh message understanding conference (MUC-7). The best result was obtained by the IE^2 system (Aone et al. 1998), which relied on manual pattern development, with an F_1 score of 76%. A close second was the 71% F_1 score of the SIFT system (S. Miller et al. 1998), which was devoid of hand-written patterns. SIFT builds an augmented parse tree of the sentence, where nodes are added to encode the semantic information conveyed by each constituent. New nodes are created using an algorithm akin to a probabilistic context-free grammar using maximum likelihood. The semantic annotations are chosen following co-occurrence counts in the training set, using dynamic programming

To put these results into perspective with latter work, note that Aone et al. (1998) mention they ran their model a 167 MHz processor with 128 MB of RAM.

S. Miller et al., "BBN: Description of the SIFT System as Used for MUC-7" MUC 1998 to search the space of augmented parse trees efficiently. SIFT also uses a model to find cross-sentence relations, which represent 10–20% of the test set. The predictions are made from a set of elemental features, one of which was whether the candidate fact was seen in a previous sample; this gives a slight aggregate orientation to SIFT, even though it is primarily a sentential approach (Section 2.1). This first systematic evaluation of models on the same dataset set the stage for the development of the relation extraction task.

Subsequently, several methods built upon carefully designed features. This is for example the case of Kambhatla (2004) who use the maximum entropy principle on the following set of features:

Kambhatla, "Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Information Extraction" ACL 2004

- entities and infix words with positional markers,
- entity types by applying NER to the corpus,
- entity levels, that is whether the entity is a composite noun or a pronoun which was linked to an entity through coreference resolution,
- the number of other words and entities appearing between e_1 and e_2,
- whether e_1 and e_2 are in the same noun phrase, verb phrase or prepositional phrase,
- the dependency neighborhood, that is the neighboring nodes in the dependency tree (see Figure 2.4),
- the syntactic path, that is the path between the entities in the syntactic parse tree (see Figure 2.5).

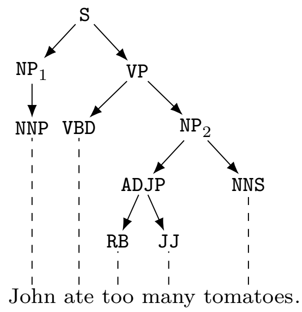


Figure 2.5: Example of syntactic parse tree generated by the PCFG parser (Klein and Manning 2003). The following abbreviations are used: `S` (simple declarative clause), `NP` (noun phrase), `VP` (verb phrase), `ADJP` (adjective phrase), `NNS` (plural noun), `NNP` (singular proper noun), `RB` (adverb), `JJ` (adjective). In contrast to a dependency tree (Figure 2.4), the words correspond to the tree's leaves, while internal nodes correspond to constituents clauses.

Let's call $(f_i(x,r))_{i\in\{1,\dots,n\}}$ the indicator functions which equal 1 iff x has feature i and convey r. The maximum entropy principle states that a classifier should match empirical data on the observed space but should have maximal entropy outside it. Calling Q^* the optimal probability model in this sense, we have:

$$\begin{aligned} Q^* &= \operatorname*{argmax}_{Q\in\mathcal{Q}} \mathrm{H}(Q) \\ &= \operatorname*{argmax}_{Q\in\mathcal{Q}} \sum_{(x,r)\in\mathcal{D}} -Q(x,r)\log Q(r\mid x) \\ &= \operatorname*{argmax}_{Q\in\mathcal{Q}} \sum_{(x,r)\in\mathcal{D}} -\hat{P}(x)Q(r\mid x)\log Q(r\mid x), \end{aligned}$$

As a reminder, $\hat{P}$ denotes the empirical distribution.

where $\mathcal{Q}$ is the set of probability mass functions matching observations:

$$\mathcal{Q} = \left\{ \text{p.m.f. } Q \;\middle|\; \mathop{\mathbb{E}}_{(x,r)\sim Q}[f_i(x,r)] = \mathop{\mathbb{E}}_{(x,r)\sim\hat{P}}[f_i(x,r)] \right\}.$$

Given this setup, the solution is part of a very restricted class of functions:

$$Q^*(r\mid x;\boldsymbol{\lambda}) \propto \exp\sum_{i=1}^{n}\lambda_i f_i(x,r).$$

The parameters $\boldsymbol{\lambda}$ are estimated using an algorithm called generalized iterative scaling (GIS, Darroch and Ratcliff 1972). Using this approach, Kambhatla (2004) evaluate their model on a dataset succeeding MUC-7 called ACE (to be precise, ACE 2003, see Section C.1 for details). They achieve an F_1 of 52.8% on 24 ACE relation subtypes.

2.3.5 Kernel Approaches

Designing a set of low-dimensional features is a tedious task: a large set of features can be computationally prohibitive, while a small set of features is necessarily limiting since they can never completely capture the essence of all samples which live in higher dimension. The kernel approaches seek to avoid this limitation by comparing samples pairwise without passing through an explicit intermediary representation. To do so, a kernel function k is defined over pair of samples:

$$k\colon (\mathcal{S} \times \mathcal{E}^2) \times (\mathcal{S} \times \mathcal{E}^2) \to \mathbb{R}_{\geq 0},$$

where k acts as a similarity measure and is required to be symmetric and positive-semidefinite. It can be shown that there is an equivalence between kernel functions and features space; for each kernel function k there is an implicit set of features $\boldsymbol{f}$ such that $k(x_1, x_2) = \boldsymbol{f}(x_1) \cdot \boldsymbol{f}(x_2)$. However, some kernel function k might be computed without having to enumerate all features $\boldsymbol{f}$.

This property is used for relation extraction by Zelenko et al. (2003) who define a similarity function k between shallow parse trees.[39] The tree kernel is defined through a similarity on nodes with a recursive call on children nodes. The equivalent feature space would need to contain all possible sub-trees which are impractical to enumerate. Zelenko et al. (2003) train a support vector machine (SVM, Cortes and Vapnik 1995) and a voted perceptron (Freund and Schapire 1999) on a dataset they hand-labeled. Culotta and Sorensen (2004) used a similar approach with a tree kernel, except that they used dependency trees (Figure 2.4) instead of syntactic parse trees. They trained SVMs on the ACE 2004 dataset (Section C.1), with their best setup reaching an F_1 of 63.2%. Finally, Zhou et al. (2005) also trained an SVM but directly used the dot product inside the feature space as a kernel.[40] Extracting a wide variety of features, they were able to reach an F_1 score of 74.7% on the ACE 2004 dataset.

Zelenko et al., "Kernel Methods for Relation Extraction" JMLR 2003

[39] A shallow parse tree is similar to a syntactic parse tree (Figure 2.5) on a partition of the words of a sentence (S. P. Abney 1991).

Culotta and Sorensen, "Dependency Tree Kernels for Relation Extraction" ACL 2004

Zhou et al., "Exploring Various Knowledge in Relation Extraction" ACL 2005

[40] In the same way that a kernel always corresponds to the dot product in a feature space, the reverse can be shown to be true too, since a Gram matrix is always semidefinite positive.

2.3.6 Piecewise Convolutional Neural Network

In the 2010s, machine learning models moved away from hand-designed features towards automatic feature extractors (Section 1.1). In relation extraction, this move was initiated by Socher et al. (2012) using an RNN-like model (Section 1.3.2), but it really started to gain traction with piecewise convolutional neural networks (PCNN, Zeng et al. 2015). PCNNs perform supervised relation extraction using deep learning. In contrast to previous models, they learn a CNN feature extractor (Section 1.3.1) on top of word2vec embeddings (Section 1.2.1) instead of using hand-engineered features. Furthermore, PCNN uses the split-in-three-affixes method of DIPRE (Figure 2.3). They feed each affix to a CNN followed by a max-pooling to obtain a fixed-length representation of the sentence, which depends on the position of the embeddings. This representation is then used to predict the relation using a linear and softmax layer. While the global position invariance of CNN is interesting for language modeling, phrases closer to entities might be of more importance for relation extraction, thus PCNN also uses temporal encoding (Section 1.3.3.2). Figure 2.6 showcases a PCNN model.

Zeng et al., "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks" EMNLP 2015

The setup described above can be used for sentential relation extraction. However, Zeng et al. (2015) and subsequent works place themselves

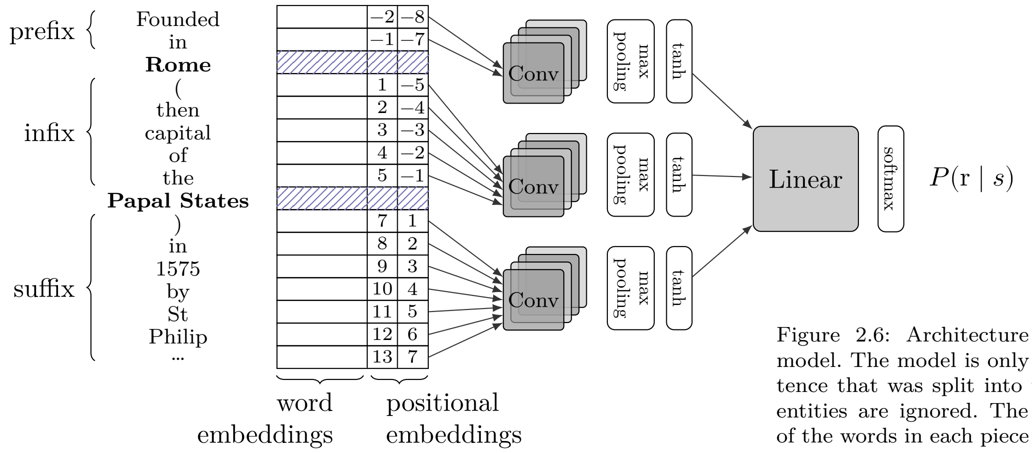


Figure 2.6: Architecture of a PCNN model. The model is only given a sentence that was split into three pieces; entities are ignored. The embeddings of the words in each piece are concatenated with two positional embeddings. Each piece is then fed to a convolutional layer, and a linear layer merges the three representations together. At the softmax output, we obtain a probability distribution over possible relations given the sentence.

in the aggregate setup. Therefore, we will wait until Section 2.4.4 to delve into the training algorithm and experimental results of PCNNs.

2.3.7 Transformer-based Models

Following the progression of Section 1.3, CNN-based models were soon replaced by transformer-based models. Soares et al. (2019) introduce the unsupervised matching the blanks (MTB) model together with an in-depth study on the use of transformers for relation extraction. We will focus on the transformer extractor in this section and study the unsupervised model in Section 2.5.6. Soares et al. (2019) introduces several methods to extract an entity-aware representation of a sentence using BERT (Section 1.3.4). These different methods can be characterized along two axes:

Soares et al., "Matching the Blanks: Distributional Similarity for Relation Learning" ACL 2019

Entity Span Identification, that is how are the entities marked in the sentence. This can be *none*, meaning that the entities are not differentiated from the other words in the sentence. It can be through *entity markers*, i.e. new tokens are introduced to mark the two entities' beginning and end, as showcased by Figures 2.12 and 2.7. Finally, it can be through a special feature of BERT: *token type embeddings*; in this case, the embeddings of the entity tokens are added to another embedding representing the slot—either e_1 or e_2—of the entity.

Output Construction, that is how a fixed-size representation is obtained from the sequence of token embeddings. A first approach is to simply use the CLS *token* embedding, i.e. the sequence's first token, which should encompass the whole sentence semantic (Section 1.3.4). A second approach is to use *entity max-pooling*: each entity is represented by the component-wise maximum along its tokens embeddings, the sentence is represented by the concatenation of its entities representations. A variant of this, using mean pooling combined with the CLS method, is used by EPGNN (Figure 2.12). These representations should better capture the semantic surrounding the entities, in contrast to the CLS token, which captures the whole sentence's semantic. Finally, a last option is to use the embeddings of the *entity start markers*; this is the option illustrated by Figure 2.7 and has the advantage to lessen the dependence of the representation on the entity surface form (Section 2.1.2 describes why this could be desirable).

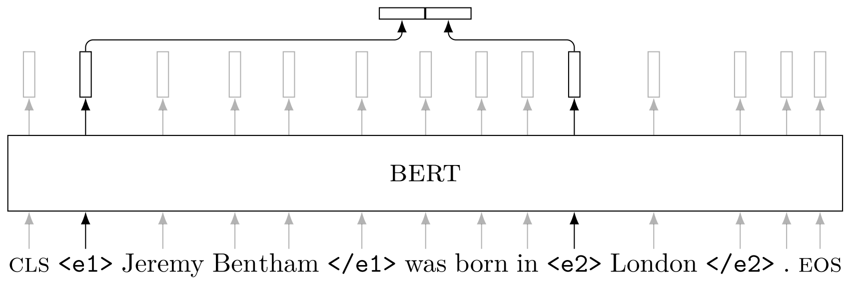


Figure 2.7: MTB entity markers–entity start sentence representation. "Bentham" was split into two subword tokens, "Ben-" and "-tham" by the BPE algorithm described in Section 1.2.3. The contextualized embeddings of most words are ignored. The final representation is only built using the representation of `<e1>` and `<e2>`. However, note that these representations are built from all the words in the sentence using an attention mechanism (Section 1.3.3). In the original work of Soares et al. (2019), the representation extracted by BERT is either fed through layer normalization (Ba et al. 2016) or to a linear layer depending on the dataset.

The best results obtained by MTB were with the entity markers–entity start method. This is the method we focus on from now on. We refer to this sentence representation model by the function BERTcoder: $\mathcal{S} \to \mathbb{R}^d$ illustrated Figure 2.7. Training is performed using a softmax layer of size $|\mathcal{R}|$ with a cross-entropy loss. Using a standard BERT-`large` pre-trained on a MLM task, MTB obtains a macro-$\overleftarrow{F_1}$ of 89.2% on the SemEval 2010 Task 8 (Section C.6).

2.4 Supervised Aggregate Extraction Models

All the approaches introduced thus far are sentential. They map each sample to a relation individually, without modeling the interactions between samples. In contrast, this section focuses on aggregate approaches (Equation 2.2). Aggregate approaches explicitly model the connections between samples. The most common aggregate method is to ensure the consistency of relations predicted for a given entity pair $\boldsymbol{e} \in \mathcal{E}^2$ by processing together all sentences $s \in \mathcal{S}$ mentioning $\boldsymbol{e}$. To this end, we define $\mathcal{D}^{\boldsymbol{e}}$ to be the dataset $\mathcal{D}$ grouped by entity pairs. Thus, instead of containing a sample $x = (s, \boldsymbol{e})$, the dataset $\mathcal{D}^{\boldsymbol{e}}$ contains bag of mentions $\boldsymbol{x} = \{(s, \boldsymbol{e}), (s', \boldsymbol{e}), \dots\}$ of the same entity pair $\boldsymbol{e}$. Most aggregate methods are built upon sentential approaches and provide a sentential assignment. Therefore, more often than not, each sample is still mapped to a relation. Therefore, the evaluations of aggregate methods follow the evaluations of sentential approaches introduced in Section 2.3.1.

2.4.1 Label Propagation

To deal with the shortage of manually labeled data, one approach is to use labels weakly correlated with the samples as in distant supervision (Section 2.2.2). Another approach is to label a small subset of the dataset but leave most samples unlabeled. This is the semi-supervised approach. The bootstrapped models (Section 2.2.1) can also be seen as semi-supervised approaches: a small number of labeled samples are given to the model, which then crawls the web to obtain new unsupervised samples. The evaluation of semi-supervised models follows the one of supervised models described in Section 2.3.1. The difference between the two lies in the fact that unsupervised samples can be used to gain a better estimate of the input distribution in the semi-supervised settings, while fully-supervised models cannot make use of unsupervised samples.

Apart from bootstrapped models, one of the first semi-supervised relation extraction systems was proposed by Chen et al. (2006). They build their model on top of hand-engineered features (Section 2.3.4) compared using a similarity function. This is somewhat similar to kernel approaches (section 2.3.5), except that this function does not need to be positive semidefinite. Given all samples in feature space, the labels from the supervised samples are propagated to the neighboring unlabeled samples using the label propagation algorithm (X. Zhu and Ghahramani 2002) listed as Algorithm 2.2. This propagation takes the form of a convex combination of other samples' labels weighted by the similarity function. Let's call sim this unlabeled sample similarity function:

$$\operatorname{sim}\colon (\mathcal{S} \times \mathcal{E}^2) \times (\mathcal{S} \times \mathcal{E}^2) \to \mathbb{R}.$$

Chen et al., "Relation Extraction Using Label Propagation Based Semi-Supervised Learning" ACL 2006

The label propagation algorithm builds a pairwise similarity matrix between labeled and unlabeled samples which have been column normalized then row normalized:

$$t_{ij} \propto \frac{\exp\left(\operatorname{sim}(x_i, x_j)\right)}{\sum\limits_{x_k \in \mathcal{D} \cup \mathcal{D}_\mathcal{R}} \exp\left(\operatorname{sim}(x_k, x_j)\right)} \quad \text{for } i, j \in \{1, \dots, |\mathcal{D}| + |\mathcal{D}_\mathcal{R}|\} \tag{2.4}$$

The relation assigned to each unlabeled sample is then recomputed by aggregating the labels—whether these labels come from $\mathcal{D}_\mathcal{R}$ or were computed at a previous iteration—of all other samples weighted by $\boldsymbol{T}$. Note that labels assigned to samples coming from $\mathcal{D}_\mathcal{R}$ are not altered. This operation is repeated until the label assignment stabilizes. This label propagation algorithm has been shown to converge to a unique solution (X. Zhu and Ghahramani 2002).

algorithm Label Propagation

Inputs: $\mathcal{D}_\mathcal{R}$ labeled dataset
$\mathcal{D}$ unlabeled dataset
Output: $\hat{\boldsymbol{r}}$ relation predictions

▷ *Initialization* ◁
$\boldsymbol{T} \leftarrow$ computed using Equation 2.4 from $\mathcal{D}_\mathcal{R}$ and $\mathcal{D}$
$\boldsymbol{Y} \leftarrow$ random stochastic matrix
for all $(s_i, \boldsymbol{e}_i, r_i) \in \mathcal{D}_\mathcal{R}$ **do**
  $y_{ij} \leftarrow \delta_{j,r_i}$
▷ *Training* ◁
loop
  $\boldsymbol{Y} \leftarrow \boldsymbol{TY}$
  for all $(s_i, \boldsymbol{e}_i, r_i) \in \mathcal{D}_\mathcal{R}$ **do**
    $y_{ij} \leftarrow \delta_{j,r_i}$
$\hat{r}_i \leftarrow \operatorname{argmax}_j y_{ij}$
output $\hat{\boldsymbol{r}}$

Algorithm 2.2: The label propagation algorithm. The notation $\delta_{a,b}$ is a Kronecker delta, equals to 1 if $a = b$ and to 0 otherwise. The two loops assigning to y_{ij} are simply enforcing that the relation assigned to the labeled samples do not deviate from their gold value.

Chen et al. (2006) tried two similarity functions: the cosine and the Jensen–Shannon of the feature vectors. They evaluated their approach on the ACE 2003 dataset (Section C.1) using different fractions of the labels to show that while their model was roughly at the same performance level than others when using the whole dataset, it decisively outperformed other methods when using a small number of labels.

2.4.2 Multi-instance Multi-label

Following the popularization of distant supervision by Mintz et al. (2009), training datasets gained in volume but lost in quality (see Section 2.2.2). In order to create models more resilient to the large number of false-positive in distantly-supervised datasets, multi-instance approaches (Dietterich et al. 1997) started to get traction.

In the article of Mintz et al. (2009), all mentions of the same entity pair are viewed as a single sample to make a prediction. Their model is a simple logistic classifier on top of hand-engineered features, which could only predict a single relation label per entity pair. However, when aggregating the features of all mentions and supervising with a single relation, Mintz et al. (2009) backpropagate to all features, i.e. the parameters used by all mentions are modified. This assumes that all mentions should convey the relation. To avoid this assumption, the more sophisticated multi-instance assumption is used:

Assumption $\mathscr{H}_{\text{MULTI-INSTANCE}}$: *All facts* $(\boldsymbol{e}, r) \in \mathcal{D}_{\text{KB}}$ *are conveyed by at least one sentence of the unlabeled dataset* $\mathcal{D}$.

$\forall (e_1, e_2, r) \in \mathcal{D}_{\text{KB}} : \exists (s, e_1, e_2) \in \mathcal{D} : (s, e_1, e_2)$ *conveys* $e_1\ r\ e_2$

MultiR (Hoffmann et al. 2011) follows such a multi-instance setup but also models multiple relations and thus does not assume $\mathscr{H}_{\text{1-ADJACENCY}}$, unlike all the models introduced thus far. Figure 2.8 illustrates this setup, which is dubbed MIML (multi-instance multi-label) following the subsequent work of Surdeanu et al. (2012).

Hoffmann et al., "Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations" ACL 2011

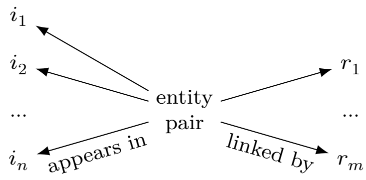


Figure 2.8: Multi-instance ($n > 1$) multi-label ($m > 1$) setup. Each entity pair appears in several instances and the two entities are linked by several relations.

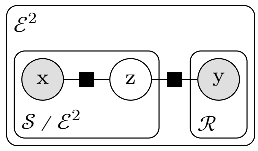


Figure 2.9: MultiR plate diagram. Where ■ denotes factor nodes.

MultiR uses a latent variable z to capture the sentential extraction. That is, for each sentence $x_i \in \mathcal{D}_\mathcal{R}$, the latent variable $\mathrm{z}_i \in \mathcal{R}$ captures the relation conveyed by x_i. Furthermore, for a given entity pair $\boldsymbol{e} \in \mathcal{E}^2$, for all $r \in \mathcal{R}$, a binary classifier y_r is used to predict whether this pair is linked by r. In this fashion, multiple relations can be predicted for the same entity pair. The model can be summarized by the plate diagram of Figure 2.9. Let's define $\mathcal{D}_\mathcal{R}^{\boldsymbol{e}}$ the dataset $\mathcal{D}_\mathcal{R}$ where samples are grouped by entity pairs. Since multiple relations can link the same entity pair, we will use $\boldsymbol{y} \in \{0, 1\}^\mathcal{R}$ to refer to the binary vector indexing the conveyed relations. Formally, MultiR defines the probability of the sentential ($\boldsymbol{z}$) and aggregate ($\boldsymbol{y}$) assignments for a mention bag ($\boldsymbol{x}$) as follow:

$$P(\boldsymbol{y}, \boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta}) \propto \prod_{r \in \mathcal{R}} \boldsymbol{\phi}^{\text{join}}(y_r, \boldsymbol{z}) \prod_{x_i \in \boldsymbol{x}} \boldsymbol{\phi}^{\text{extract}}(z_i, x_i; \boldsymbol{\theta}) \tag{2.5}$$

where $\boldsymbol{\phi}^{\text{join}}$ simply aggregate the predictions for all mentions:

$$\boldsymbol{\phi}^{\text{join}}(y_r, \boldsymbol{z}) = \begin{cases} 1 & \text{if } y_r = 1 \land \exists i : z_i = r \\ 0 & \text{otherwise} \end{cases}$$

and $\boldsymbol{\phi}^{\text{extract}}$ is a weighted sum of several hand-designed features:

$$\boldsymbol{\phi}^{\text{extract}}(z_i, x_i; \boldsymbol{\theta}) = \exp\left(\sum_{\text{feature } j} \theta_j \phi_j(z_i, x_i)\right)$$

We now describe the training algorithm used by MultiR, which is listed as Algorithm 2.3. Following the multi-instance setup, MultiR assumes that every fact $(e_1, r, e_2) \in \mathcal{D}_{\text{KB}}$ is conveyed by at least one mention $(s, e_1, e_2) \in \mathcal{D}$. This can be seen in the first product of Equation 2.5: if a single gold relation is not predicted for any sentence, the whole probability mass function drops to 0. This means that during inference, each relation r conveyed in the knowledge base must be covered by at least one sentential extraction z. Given all sentences $\boldsymbol{x}_i \subseteq \mathcal{D}$ containing an entity pair (e_1, e_2), when the model does not predict the actual set of relations $\boldsymbol{y}_i = \{\, r \mid (e_1, r, e_2) \in \mathcal{D}_{\text{KB}} \,\}$, the parameters $\boldsymbol{\theta}$ must be tuned such that every relation $r \in \boldsymbol{y}_i$ is conveyed by at least one sentence, as expressed by the line:

$$\boldsymbol{z}^* \leftarrow \operatorname*{argmax}_{\boldsymbol{z}} P(\boldsymbol{z} \mid \boldsymbol{x}_i, \boldsymbol{y}_i; \boldsymbol{\theta}).$$

In particular, note that if an entity pair is linked by more relations than it has mentions in the text, the algorithm collapses since each mention conveys a single relation.

This can be reframed as a weighted edge-cover problem, where the edge weights are given by $\boldsymbol{\phi}^{\text{extract}}(z_i, x_i; \boldsymbol{\theta})$. The MultiR training algorithm can be seen as maximizing the likelihood $P(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\theta})$ where a Viterbi approximation was used—the expectations being replaced with maxima.

The multi-instance multi-label (MIML) phrase was introduced by Surdeanu et al. (2012). Their approach is similar to that of MultiR except that they train a classifier for $\boldsymbol{\phi}^{\text{join}}$ instead of using a deterministic process. Their training procedure also differs. They train in the Bayesian framework using an expectation–maximization algorithm. In general, MIML approaches are challenging to evaluate systematically since they suffer from

Surdeanu et al., "Multi-instance Multi-label Learning for Relation Extraction" EMNLP 2012

algorithm MultiR

Input: $\mathcal{D}^{\mathbf{e}}_{\mathcal{R}}$ a supervised multi-instance dataset
Output: $\boldsymbol{\theta}$ model parameters

$\boldsymbol{\theta} \leftarrow \mathbf{0}$
loop
  for all $(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}^{\mathbf{e}}_{\mathcal{R}}$ **do**
    $(\boldsymbol{y}', \boldsymbol{z}') \leftarrow \operatorname{argmax}_{\boldsymbol{y},\boldsymbol{z}} P(\boldsymbol{y}, \boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta})$
    if $\boldsymbol{y}' \neq \boldsymbol{y}_i$ **then**
      $\boldsymbol{z}^* \leftarrow \operatorname{argmax}_{\boldsymbol{z}} P(\boldsymbol{z} \mid \boldsymbol{x}_i, \boldsymbol{y}_i; \boldsymbol{\theta})$
      $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{z}^*) - \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{z}')$
output $\boldsymbol{\theta}$

Algorithm 2.3: The MultiR training algorithm. For each bag of mentions $\boldsymbol{x}_i$, the more likely sentential and aggregate predictions $(\boldsymbol{y}', \boldsymbol{z}')$ are made. If the predicted relations are different from the true relations $\boldsymbol{y}_i$ linking the two entities, the parameters $\boldsymbol{\theta}$ are adjusted such that $\boldsymbol{z}$ cover all relations in $\boldsymbol{y}_i$.

low precision due to incomplete knowledge bases. In particular, they were not compared with traditional supervised approaches. For reference, Surdeanu et al. (2012) compare the three methods mentioned in this section on the same datasets and observe that at the threshold at which recall goes over 30%, the precision falls under 30%.

2.4.3 Universal Schemas

Another important weakly-supervised model is the universal schema approach designed by Riedel et al. (2013). In their setting, existing relations and surface forms linking two entities are considered to be of the same nature. Slightly departing from their terminology, we refer to the union of relations ($\mathcal{R}$) and surface forms ($\mathcal{S}$) by the term "items" ($\mathcal{I} = \mathcal{R} \cup \mathcal{S}$) for their similarity with the collaborative filtering concept. Riedel et al. (2013) consider that entity pairs are linked by items such that the dataset available could be refered to as $\mathcal{D}_{\mathcal{I}} \subseteq \mathcal{E}^2 \times \mathcal{I}$. This can be obtained by taking the union of an unlabeled dataset $\mathcal{D}$ and a knowledge base $\mathcal{D}_{\text{KB}}$. This dataset $\mathcal{D}_{\mathcal{I}}$ can be seen as a matrix with entity pairs corresponding to rows and items corresponding to columns. With this in mind, relation extraction resembles collaborative filtering. Figure 2.10 gives an example of this matrix that we will call $\boldsymbol{M} \in \mathbb{R}^{\mathcal{E}^2 \times \mathcal{I}}$.

Riedel et al., "Relation Extraction with Matrix Factorization and Universal Schemas" NACL 2013

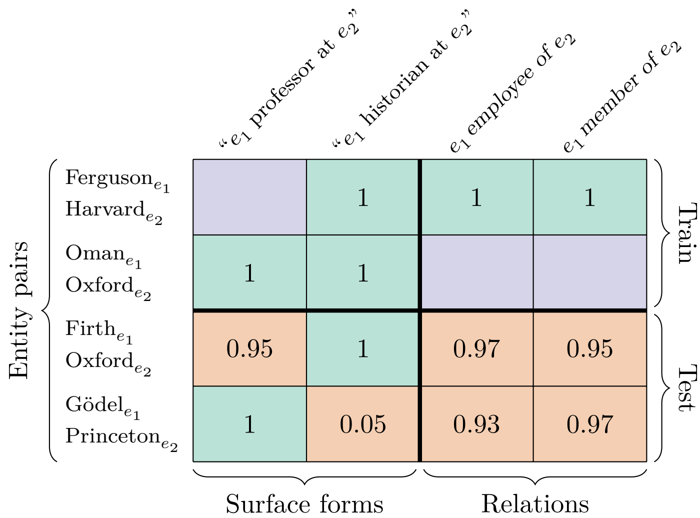


Figure 2.10: Universal schema matrix. Observed entity–item pairs are shown in green, blue cells are unobserved values, while orange cells are unobserved values for which a prediction was made. The observed values on the left (surface forms) come from an unsupervised dataset $\mathcal{D}$, while the observed values on the right (relations) come from a knowledge base $\mathcal{D}_{\text{KB}}$.

Riedel et al. (2013) purpose to model this matrix using a combination of three models. One of them being a low-rank matrix factorization:

$$m^{\text{F}}_{ei} = \sum_{j=0}^{d} u_{ej} v_{ij}$$

where d is a hyperparameter, and $\boldsymbol{U} \in \mathbb{R}^{\mathcal{E}^2 \times d}$ and $\boldsymbol{V} \in \mathbb{R}^{\mathcal{I} \times d}$ are model parameters. The two other models are an inter-item neighborhood model and selectional preferences (described in Section 1.4.2.1), which we do not detail here. Training such a model is difficult since we do not have access to negative facts: not observing a sample $(\boldsymbol{e}, i) \notin \mathcal{D}_{\mathcal{I}}$ does not necessarily imply that this sample is false. To cope with this issue, Riedel et al. (2013) propose to use the Bayesian personalized ranking model (BPR, Rendle et al. 2009). Instead of enforcing each element m_{ei} to be equal to 1 or 0, BPR relies upon a ranking objective pushing element observed to be true to be ranked higher than unobserved elements. This is done through a contrastive objective between observed positive samples and unobserved negative samples from a uniform distribution:

Rendle et al., "BPR: Bayesian Personalized Ranking from Implicit Feedback" UAI 2009

$$J_{\text{US}}(\boldsymbol{\theta}) = \sum_{(\boldsymbol{e}^+, i) \in \mathcal{D}_{\mathcal{I}}} \sum_{\substack{(\boldsymbol{e}^-, i) \in \mathcal{E}^2 \times \mathcal{I} \\ (\boldsymbol{e}^-, i) \notin \mathcal{D}_{\mathcal{I}}}} \log \sigma(m_{e^+ i} - m_{e^- i})$$

This objective can be directly maximized using stochastic gradient ascent. Riedel et al. (2013) experiment on a NYT + FB dataset, this means the unsupervised dataset $\mathcal{D}$ comes from the New York Times (NYT, Section C.5) and the knowledge base $\mathcal{D}_{\text{KB}}$ is Freebase (FB, Section C.3).

2.4.4 Aggregate PCNN Extraction

PCNN is a sentence-level feature extractor introduced in Section 2.3.6. Zeng et al. (2015) introduce the PCNN feature extractor together with a multi-instance learning algorithm. Given a bag of mentions $\boldsymbol{x} \in \mathcal{D}^{\boldsymbol{e}}$, for each mention $x_i \in \boldsymbol{x}$, they model $P(\text{r} \mid x_i; \boldsymbol{\theta})$. However, the optimization is done over each bag of mentions separately:

Zeng et al., "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks" EMNLP 2015

$$\mathcal{L}_{\text{PCNN}}(\boldsymbol{\theta}) = -\sum_{(\boldsymbol{x}, r) \in \mathcal{D}^{\boldsymbol{e}}_{\mathcal{R}}} \log P(r \mid x^*; \boldsymbol{\theta}) \tag{2.6}$$

$$x^* = \operatorname*{argmax}_{x_i \in \boldsymbol{x}} P(r \mid x_i; \boldsymbol{\theta}) \tag{2.7}$$

In other words, for a set of mention $\boldsymbol{x}$ of an entity pair, the network back-propagates only on the sample that predicts a relation with the highest certainty. Thus PCNN is a multi-instance single-relation model, it assumes $\mathscr{H}_{\text{MULTI-INSTANCE}}$ but also $\mathscr{H}_{\text{1-ADJACENCY}}$.

Zeng et al. (2015) continue to use the experimental setup of Surdeanu et al. (2012), i.e. using a distantly supervised dataset, but complement it with a manual evaluation to have a better estimate of the precision.

Y. Lin et al. (2016) improve the PCNN model with an attention mechanism over mentions to replace the argmax of Equation 2.7. The attention mechanism's memory is built from the output of the PCNN on each mention without applying a softmax; the PCNN is simply used to produce a representation for each mention. Equations 2.6 and 2.7 are then replaced

Y. Lin et al., "Neural Relation Extraction with Selective Attention over Instances" ACL 2016 by:

$$\mathcal{L}_{\text{Lin}}(\boldsymbol{\theta}) = -\sum_{(\boldsymbol{x},r)\in\mathcal{D}_{\mathcal{R}}^{\boldsymbol{e}}} \log P(r \mid \boldsymbol{x};\boldsymbol{\theta})$$

$$P(r \mid \boldsymbol{x};\boldsymbol{\theta}) \propto \exp(\boldsymbol{W}\boldsymbol{s}(\boldsymbol{x},r)+\boldsymbol{b})$$

$$\boldsymbol{s}(\boldsymbol{x},r) = \sum_{x_i\in\boldsymbol{x}} \alpha_i \,\text{PCNN}(x_i)$$

where the α_i are attention weights computed from a bilinear product between the query r and the memory $\text{PCNN}(\boldsymbol{x})$, similarly to the setup of Section 1.3.3. Y. Lin et al. (2016) show that this modification improves the results of PCNN, this can be seen as a relaxation of $\mathscr{H}_{\text{MULTI-INSTANCE}}$: the standard PCNN approach assumes that each fact in $\mathcal{D}_{\text{KB}}$ is conveyed by a single sentence through its argmax; in contrast, the attention approach simply assumes that all facts are conveyed in $\mathcal{D}$, at least by one sentence but possibly by several ones.

2.4.5 Entity Pair Graph

The multi-instance approach shares information at the entity pair level. However, information could also be shared between different entity pairs. This is the idea put forth by entity pair graph neural network (EPGNN, Zhao et al. 2019). The basic sharing unit becomes the entity: when two mentions $(s, e_1, e_2), (s', e'_1, e'_2) \in \mathcal{D}$ share at least one entity ($\{e_1, e_2\} \cap \{e'_1, e'_2\} \neq \emptyset$), their features interact with each other in order to make a prediction. The sharing of information is made following an entity pair graph that links together bags of mentions with a common entity as illustrated in Figure 2.11.

Zhao et al., "Improving Relation Classification by Entity Pair Graph" PMLR 2019

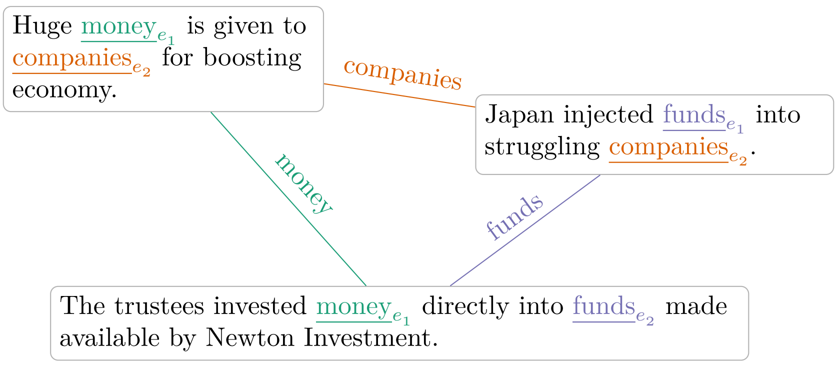


Figure 2.11: Entity pair graph. Each node corresponds to a bag of mentions, each edge of the graph corresponds to an entity in common between the two bags, the edges are labeled with the shared entity. For illustration purpose, we show a single sample per bag. This example is from the SemEval 2010 Task 8 dataset (described in Section C.6). All sentences convey the *entity-destination* relation.

To obtain a distributed representation for a sentence, EPGNN uses BERT (Section 1.3.4). More precisely, it combines the embedding of the CLS token[41] with the embeddings corresponding to the two entities through a mean pooling. The sentence feature extraction architecture is illustrated by Figure 2.12. This is one of several methods to obtain an entity-aware fixed-size representation of a tagged sentence; other approaches are developed in Section 2.3.7.

[41] As a reminder, the CLS token is the marker for the beginning of the sentence, its embedding purposes to represent the whole sentence.

Given a vector representation for each sentence in the dataset, we can label the vertices of the entity pair graph. A spectral graph convolutional network (GCN, Section 4.3.2) is then used to aggregate the information of its neighboring samples into each vertex. Thus, EPGNN produces two representations for a sample: one sentential and one topological. From

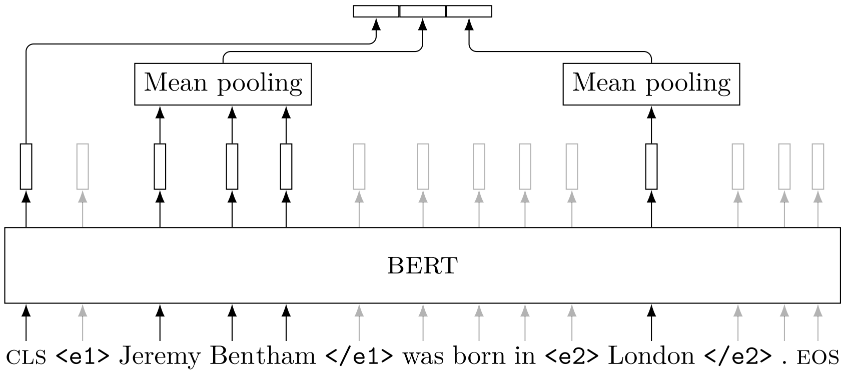


Figure 2.12: EPGNN sentence representation. "Bentham" was split into two subword tokens, "Ben-" and "-tham" by the BPE algorithm described in Section 1.2.3. The contextualized embeddings of most words are ignored. The final representation is only built using the entities span and the CLS token. Not appearing on the figure are linear layers used to post-process the output of the mean poolings and the final representation as well as a ReLU nonlinearity. Compare to Figure 2.7.

these two representations, a prediction is made using a linear and softmax layer. Since a single relation is produced for each sample, EPGNN is trained using the usual classification cross-entropy loss. More details on graph-based approaches are given in Chapter 4.

Zhao et al. (2019) evaluate EPGNN on two datasets, SemEval 2010 Task 8 (Section C.6) and ACE 2005 (Section C.1). Reaching a half-directed macro-$\overleftarrow{F_1}$ of 90.2% on the first one, and a micro-F_1 of 77.1% on the second.

2.5 Unsupervised Extraction Models

> *If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake.*
> — Yann LeCun, Inaugural Lecture at Collège de France (2016)

In the unsupervised setting, no samples are labeled with a relation, i.e. all samples are triplets (sentence, head entity, tail entity) from $\mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}^2$. Furthermore, no information about the relation set $\mathcal{R}$ is available. This is problematic since whether a specific semantic link is worthy of appearing in $\mathcal{R}$ or not is not well defined. Having so little information about what constitutes a relation makes the problem intractable if we do not impose some restrictions upon $\mathcal{R}$. All unsupervised models presented in this section are not universal and make some kind of assumption on the structure of the data or on its underlying knowledge base. However, developing unsupervised relation extraction models is still interesting for three reasons: they (1) do not necessitate labeled data except for validating the models; (2) can uncover new relation types; and (3) can be trained from large unlabeled datasets and then fine-tuned for specific relations.

For all models, we list the important modeling hypothesis such as $\mathscr{H}_{\text{1-ADJACENCY}}$ and $\mathscr{H}_{\text{PULLBACK}}$ introduced previously. Appendix B contains a list of assumptions with some counterexamples and references to the sections where they were introduced. We strongly encourage the reader to refer to it, especially when the implications of a modeling hypothesis is not immediately clear.

2.5.1 Evaluation

The output of unsupervised models vary widely. The main modus operandi can be categorized into two categories:

Clustering A first approach is to cluster the samples such that all samples in the same cluster convey the same relation and samples in different clusters convey different relations.

Similarity Space A second approach is to associate each sample with an element of a vector space equipped with a similarity function. If two samples are similar in this vector space, they convey similar relations. This can be seen as a soft version of the clustering approach.

This distinction has an impact on how we evaluate the models. In the first case, standard clustering metrics are used. We introduce B^3 (Bagga and Baldwin 1998), V-measure (Rosenberg and Hirschberg 2007) and ARI (Hubert and Arabie 1985) in Section 2.5.1.1. They are the most prevalent metrics in cluster evaluation, B^3 in particular is widely used in unsupervised relation extraction. In the second case, a few-shot evaluation can be used (Han et al. 2018). We introduce this approach in Section 2.5.1.2.

A difficulty of evaluating unlabeled clusters is that we do not know which cluster should be compared to which relation. A possible solution to this problem is to use a small number of labeled samples, which can be used to constrain the output of a model to fall into a specific relation set $\mathcal{R}$. This setup is actually similar to semi-supervised approaches such as label propagation (Section 2.4.1), except that the model must be trained in an unsupervised fashion before being fine-tuned on the supervised dataset. Similar to the label propagation model evaluation, unsupervised models evaluated by fine-tuning on a supervised dataset usually report performance varying the number of train labels. These performances are measured using the standard supervised metrics introduced in Section 2.3.1. Evaluating performances as a pre-training method can be used for all unsupervised models, in particular similarity-space-based approaches.

2.5.1.1 Clustering Metrics

In this section, we describe three metrics used to evaluate clustering approaches. The first metric, B^3 was first introduced to unsupervised relation extraction by rel-LDA (Yao et al. 2011, Section 2.5.4), while the other two were proposed as complements by Simon et al. (2019) presented in Chapter 3.

> *The cake is a lie.*
> — Valve, "Portal" (2007)

To clearly describe these different clustering metrics, we propose a common probabilistic formulation—in practice, these probabilities are estimated on the validation and test sets—and use the following notations. Let X and Y be random variables corresponding to samples in the dataset. Following Section 2.3.1, we denote by $c(\mathrm{X})$ the predicted cluster of X and $g(\mathrm{X})$ its conveyed gold relation.[42]

[42] This implies that a labeled dataset is sadly necessary to evaluate an unsupervised clustering model.

B^3 The metric most commonly computed for unsupervised model evaluation is a generalization of F_1 for clustering tasks called B^3 (Bagga and Baldwin 1998). The B^3 precision and recall are defined as follows:

Bagga and Baldwin, "Entity-Based Cross-Document Coreferencing Using the Vector Space Model" ACL 1998

$$\mathrm{B}^3\,\mathrm{precision}(g, c) = \mathop{\mathbb{E}}_{\mathrm{X},\mathrm{Y}\sim\mathcal{U}(\mathcal{D}_\mathcal{R})} P(g(\mathrm{X}) = g(\mathrm{Y}) \mid c(\mathrm{X}) = c(\mathrm{Y}))$$
$$\mathrm{B}^3\,\mathrm{recall}(g, c) = \mathop{\mathbb{E}}_{\mathrm{X},\mathrm{Y}\sim\mathcal{U}(\mathcal{D}_\mathcal{R})} P(c(\mathrm{X}) = c(\mathrm{Y}) \mid g(\mathrm{X}) = g(\mathrm{Y}))$$

As precision and recall can be trivially maximized by putting each sample in its own cluster or by clustering all samples into a single class, the main metric B^3 F_1 is defined as the harmonic mean of precision and recall:

$$\mathrm{B}^3 F_1(g, c) = \frac{2}{\mathrm{B}^3\,\mathrm{precision}(g, c)^{-1} + \mathrm{B}^3\,\mathrm{recall}(g, c)^{-1}}$$

While the usual precision (Section 2.3.1) can be seen as the probability that a sample with a given prediction is correct, the B^3 precision cannot use the correct relation as a reference to determine the correctness of a prediction. Instead, whether an assignment is correct is computed as the expectation that a sample is accurately classified relatively to all other samples grouped in the same cluster.

V-measure Another metric is the entropy-based V-measure (Rosenberg and Hirschberg 2007). This metric is defined by homogeneity and completeness, which are akin to B^3 precision and recall but rely on conditional entropy. For a cluster to be homogeneous, we want most of its elements to convey the same gold relation. In other words, the distribution of gold relations inside a cluster must have low entropy. This entropy is normalized by the unconditioned entropy of the gold relations to ensure that it does not depend on the size of the dataset:

Rosenberg and Hirschberg, "V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure" EMNLP 2007

$$\text{homogeneity}(g, c) = 1 - \frac{\mathrm{H}\left(c(\mathrm{X}) \mid g(\mathrm{X})\right)}{\mathrm{H}\left(c(\mathrm{X})\right)}.$$

Similarly, for a cluster to be complete, we want all the elements conveying the same gold relation to be captured by this cluster. In other words, the distribution of clusters inside a gold relation must have low entropy:

$$\text{completeness}(g, c) = 1 - \frac{\mathrm{H}\left(g(\mathrm{X}) \mid c(\mathrm{X})\right)}{\mathrm{H}\left(g(\mathrm{X})\right)}.$$

As B^3, the V-measure is summarized by the F_1 value:

$$\text{V-measure}(g, c) = \frac{2}{\text{homogeneity}(g, c)^{-1} + \text{completeness}(g, c)^{-1}}.$$

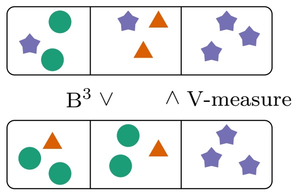


Figure 2.13: Comparison of B^3 and V-measure. Samples conveying three different relations indicated by shape and color are clustered into three boxes. The two rows represent two different clusterings, B^3 favors the first one while V-measure favors the second. V-measure prefers the second clustering since the blue star cluster is kept pure; on the other hand, the green circle cluster is impure no matter what, so its purity is not taken as much into account by the V-measure compared to B^3.

Compared to B^3, the V-measure penalizes small impurities in a relatively "pure" cluster more harshly than in less pure ones. Symmetrically, it penalizes a degradation of a well-clustered relation more than of a less-well-clustered one. This difference is illustrated in Figure 2.13.

Adjusted Rand Index The Rand index (RI, Rand 1971) is the last clustering metric we consider, it is defined as the probability that cluster and gold assignments are compatible:

$$\mathrm{RI}(g, c) = \mathop{\mathbb{E}}_{\mathrm{X},\mathrm{Y}}\left[P(c(\mathrm{X}) = c(\mathrm{Y}) \Leftrightarrow g(\mathrm{X}) = g(\mathrm{Y}))\right]$$

In other words, given two samples, the RI is improved when both samples are in the same cluster and convey the same gold relation or when both samples are in different clusters and convey different relations; otherwise, the RI deteriorates. The adjusted Rand index (ARI, Hubert and Arabie 1985) is a normalization of the Rand index such that a random assignment has an ARI of 0, and the maximum is 1:

Hubert and Arabie, "Comparing partitions" JOC 1985

$$\mathrm{ARI}(g, c) = \frac{\mathrm{RI}(g, c) - \mathop{\mathbb{E}}_{c \sim \mathcal{U}(\mathcal{R}^\mathcal{D})}[\mathrm{RI}(g, c)]}{\max_{c \in \mathcal{R}^\mathcal{D}} \mathrm{RI}(g, c) - \mathop{\mathbb{E}}_{c \sim \mathcal{U}(\mathcal{R}^\mathcal{D})}[\mathrm{RI}(g, c)]}$$

In practice, the ARI can be computed from the elements of the confusion matrix. Compared to the previous metrics, ARI will be less sensitive to a discrepancy between precision–homogeneity and recall–completeness since it is not a harmonic mean of both.

2.5.1.2 Few-shot

Clustering metrics are problematic since producing a clustering with no a priori knowledge on the relation schema $\mathcal{R}$ leads to unsolvable problems:

- Should the relation *sibling* be cut into *brother* and *sister*?
- Is the relation between a country and its capital the same as the one between a county and its seat?
- Is the ear *part of* the head in the same fashion that the star Altair is *part of* the Aquila constellation?

This section only presents Few-shot evaluation. It is possible—and quite common—to train a model using a few-shot objective, usually as a fine-tuning phase before a few-shot evaluation. Since we are mostly interested in unsupervised approaches, we do not delve into few-shot training. See Han et al. (2018) for details.

All of these questions can be answered differently depending on the design of the underlying knowledge base. However, unsupervised clustering algorithms do not depend on $\mathcal{R}$. They must decide whether "Phaedra is the sister of Ariadne" and "Castor is the brother of Pollux" go inside the same cluster independently of these design choices.

Fine-tuning on a supervised dataset solves this problem but adds another. The evaluation no longer assesses the proficiency of a model to learn from unlabeled data alone; it also evaluates its ability to adapt to labeled samples. Furthermore, the smaller the labeled dataset is, the more results have high variance. On the other hand, the larger the labeled dataset is, the less the experiment evaluates the unsupervised phase.

A few-shot evaluation can be used to answer these caveats. Instead of evaluating a clustering of the samples, few-shot experiments evaluate a similarity function between samples: $\operatorname{sim}\colon \mathcal{D} \times \mathcal{D} \to \mathbb{R}$. Given a query sample $x^{(q)}$ and a set of candidates $\boldsymbol{x}^{(c)} = \{x_i^{(c)} \mid i = 1, \dots, C\}$, the model is evaluated on whether it is able to find the candidate conveying the same relation as the query. This is simply reported as an accuracy by comparing $\operatorname{argmax}_{x \in \boldsymbol{x}^{(c)}} \operatorname{sim}(x^{(q)}, x)$ with the correct candidate.

C is the number of candidates, in Table 2.2 we have $C = 5$.

Query:
It flows into the Hörsel$_{e_2}$ in Eisenach$_{e_1}$.
Candidates:
It is remake of Hindi$_{e_2}$ film "Tezaab$_{e_1}$".
Cynidr$_{e_1}$ was the son of St Gwladys$_{e_2}$.
→ Herron Island$_{e_1}$ lies in Case Inlet$_{e_2}$.
He gained the support of Admiral$_{e_2}$ Edward Russell$_{e_1}$.
NGC 271$_{e_1}$ is a spiral galaxy in the constellation Cetus$_{e_2}$.

Table 2.2: Few-shot problem. For ease of reading, the entity identifiers—such as Q450036 for "Hörsel"—are not given. Both the query and the third candidate convey the relation P206 *located in or next to body of water*.

Table 2.2 gives an example of a few-shot problem. It illustrates the five-way one-shot problem, meaning that we must choose a relation among five and that each of the five relations is represented by a single sample. Another popular variant is the ten-way five-shot problem: the candidates are split into ten bags of five samples each, all samples in a bag convey the same relation, and the goal is to predict the bag in which the query belongs. Candidates are sometimes referred to as "train set" and the query as "test set" since this can be seen as an extremely small dataset with five training samples and one test sample.

Quite confusingly, they can also be referred to as "meta-train" and "meta-test." Indeed, to follow the usual semantic of the "meta-" prefix, the "meta-sets" should refer to sets of (query, candidates) tuples, not the candidates themselves.

FewRel, described in Section C.2, is the standard few-shot dataset. In FewRel, Altair is not P361 *part of* Aquila, it is P59 *part of constellation* Aquila. However, this design decision does not influence the evaluation. Given the query "Altair is located in the Aquila constellation," a model ought to rank this sample as more similar to samples conveying *part of constellation* than to those conveying other kinds of *part of* relationships. If FewRel made the opposite design choice, the model would still be able to achieve high accuracy by ensuring *part of* samples are similar. The decision to split or not the *part of* relation should be of no concern to the unsupervised model.

2.5.2 Open Information Extraction

In Open information extraction (OIE, Banko et al. 2007), the closed-domain assumption (Section 2.1.1.2) is neither made for relations nor entities, which are extracted jointly. Instead $\mathcal{E}$ and $\mathcal{R}$ are implicitly defined from the language itself, typically a fact (e_1, r, e_2) is expressed as a triplet such as (noun phrase, verb phrase, noun phrase). This makes OIE particularly interesting when processing large amounts of data from the web, where there can be many unanticipated relations of interest.

Banko et al., "Open Information Extraction from the Web" IJCAI 2007

This section focuses on TextRunner, the first model implementing OIE. It uses an aggregate extraction setup where $\mathcal{D}$ is directly mapped to $\mathcal{D}_{\text{KB}}$, with the peculiarity that $\mathcal{D}_{\text{KB}}$ is defined using surface forms only. The hypothesis on which TextRunner relies is that the surface form of the relation conveyed by a sentence appears in the path between the two entities in its dependency tree. In the OIE setup, these surface forms can then be used as labels for the conveyed relations, thereby using the language itself as the relation domain $\mathcal{R}$. TextRunner can be split into three parts:

The Learner is a naive Bayes classifier, trained on a small dataset to predict whether a fact (e_1, r, e_2) is trustworthy. To extract a set of samples for this task, a dependency parser (Figure 2.4) is run on the dataset and tuples (e_1, r, e_2) are extracted where e_1 and e_2 are base noun phrases and r is the dependency path between the two entities. The tuples are then automatically labeled as trustworthy or not according to a set of heuristics such as the length of the dependency path and whether it crosses a sentence boundary. The naive Bayes classifier is then trained to predict the trustworthiness of a tuple given a set of hand-engineered features (Section 2.3.4).

The Extractor extracts trustworthy facts on the whole dataset. The features on which the Learner is built only depend on part-of-speech (POS) tags (noun, verb, adjective…) such that the Extractor does not need to run a dependency parser on all the sentences in the entire dataset. While the Learner uses the dependency path for r, the Extractor uses the infix from which non-essential phrases (such as adverbs) are eliminated heuristically. Thus the Extractor simply runs a POS tagger on all sentences, finds all possible entities e, estimates a probable relation r and filters them using the Learner to output a set of trustworthy facts.

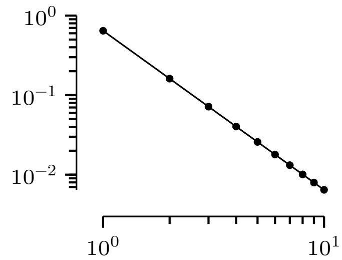
Dependency parsers tend to be a lot slower than POS taggers.

The Assessor assigns a probability that a fact is true from redundancy in the dataset using the urns model of Downey et al. (2005). This model uses a binomial distribution to model the probability that a correct fact appears k times among n extractions with a fixed repetition rate. Furthermore, it assumes both correct and incorrect facts follow different Zipf's laws. The shape parameter s_I of the distribution of incorrect facts is assumed to be 1. While the shape parameter s_C of the distribution of correct facts as well as the number of correct facts N_C are estimated using an expectation–maximization algorithm. In the expectation step, the binomial and Zipf distribution assumptions can be combined using Bayes' theorem to estimate whether a fact is correct or not. In the maximization step, the parameters s_C and N_C are estimated.

Zipf's law comes from the externalist linguistic school. It follows from the observation that the frequency of the second most common word is half the one of the most frequent word, that the one of the third most common word is a third of the one of the most frequent, etc. The same distribution can often be observed in information extraction. Zipf's law is parametrized by a shape s and the number of elements N:

$$P(x \mid s) \propto \begin{cases} x^{-s} & \text{for } x \in \{1, \dots, N\} \\ 0 & \text{otherwise} \end{cases}$$

A Zipf's law is easily recognizable on a log–log scale, its probability mass function being a straight line. Take for example the Zipf's law with parameters $s = 2$ and $N = 10$:



Banko et al. (2007) compare their approach to KnowItAll, an earlier work similar to OIE but needing a list of relations (surface forms) as input to define the target relation schema $\mathcal{R}$. On a set of ten relations, they manually labeled the extracted facts as correct or not, obtaining an error rate of 12% for TextRunner and 18% for KnowItAll. They further run their model on 9 million web pages, extracting 7.8 million facts.

A limitation of the OIE approach is that it heavily depends on the raw surface form and suffers from bad generalization. The two facts "Bletchley Park *known as* Station X" and "Bletchley Park *codenamed* Station X" are considered different by TextRunner since the surface forms conveying the relations in the underlying sentences are different. Subsequent OIE approaches try to address this problem, such as Yates et al. (2007), which extend TextRunner with a resolver (Yates and Etzioni 2007) to merge synonyms. However, this problem is not overcome yet and is still an active area of research. Furthermore, since the input of OIE systems is often taken to be the largest possible chunk of the web, and since the extracted facts do not follow a strict nomenclature, a fair evaluation of OIE systems among themselves or to other unsupervised relation extraction models is still not feasible.

2.5.3 Clustering Surface Forms

The first unsupervised relation extraction model was the clustering approach of Hasegawa et al. (2004). It is somewhat similar to DIRT (Section 2.3.3) in that it uses a similarity between samples. However, their work goes one step further by using this similarity to build relation classes. Furthermore, Hasegawa et al. (2004) does not assume $\mathscr{H}_{\text{PULLBACK}}$, i.e. it does not assume that the sentence and entities convey the relation separately, on their own. Instead, its basic assumption is that the infix between two entities is the expression of the conveyed relation. As such, if two infixes are similar, the sentences convey similar relations. Furthermore, NER (see the introduction of Chapter 2) is performed on the text instead of simple entity chunking. This means that all entities are tagged with a type such as "organization" and "person." These types strongly constrain the relations through the following assumption:

Hasegawa et al., "Discovering Relations among Named Entities from Large Corpora" ACL 2004

As a reminder, the infix is the span of text between the two entities in the sentence.

Assumption $\mathscr{H}_{\text{TYPE}}$: *All entities have a unique type, and all relations are left and right restricted to one of these types.*

$$\exists \mathcal{T} \text{ partition of } \mathcal{E} : \forall r \in \mathcal{R} : \exists X, Y \in \mathcal{T} : r \bullet \breve{r} \cup \mathbf{1}_X = \mathbf{1}_X \land \breve{r} \bullet r \cup \mathbf{1}_Y = \mathbf{1}_Y$$

Following Section 1.4.1, $\breve{r}$ is the converse relation of r, i.e. the relation with e_1 and e_2 in the reverse order. $\bullet$ is the composition operator and $\mathbf{1}_X$ the complete relation over X. $r \bullet \breve{r}$ is the relation linking all the entities which appear as subject (e_1, on the left hand side) of r to themselves. This relation is constrained to be between entities in X. Less relevant to this formula, $r \bullet \breve{r}$ also links together entities linked by r to the same object.

This is a natural assumption for many relations; for example, the relation *born in* is always between a person and a geopolitical entity (GPE).

Given a pair of entities $(e_1, e_2) \in \mathcal{E}^2$, Hasegawa et al. (2004) collect all samples in which they appear and extract a single vector representation from all these samples. This representation is built from the bag of words of the infixes weighted by TF–IDF (term frequency–inverse document frequency). Since a bag of words discards the ordering of the words or entities,

Here, we assume that the partition $\mathcal{T}$ is not degenerate and somewhat looks like a standard NER classification output. Otherwise, $\mathcal{T} = \{\mathcal{E}\}$ is a valid partition of $\mathcal{E}$, and this assumption is tautological.

the variant of TF–IDF used takes into account the directionality:

$$\begin{aligned}\text{TF}(w, e_1, e_2) =& \text{number of times } w \text{ appears between } e_1 \text{ and } e_2 \\ & - \text{number of times } w \text{ appears between } e_2 \text{ and } e_1 \\ \text{IDF}(w) =& (\text{number of documents in which } w \text{ appears})^{-1} \\ \text{TF–IDF}(w, e_1, e_2) =& \text{TF}(w, e_1, e_2) \cdot \text{IDF}(w)\end{aligned}$$

From this definition we obtain a representation $\boldsymbol{z}_{e_1,e_2} \in \mathbb{R}^V$ of the pair $(e_1, e_2) \in \mathcal{E}^2$ by taking the value of TF–IDF(w, e_1, e_2) for all $w \in V$. Given two entity pairs, their similarity is defined as follow:

$$\text{sim}(\boldsymbol{e}, \boldsymbol{e}') = \cos(\boldsymbol{z}_{\boldsymbol{e}}, \boldsymbol{z}_{\boldsymbol{e}'}) = \frac{\boldsymbol{z}_{\boldsymbol{e}} \cdot \boldsymbol{z}_{\boldsymbol{e}'}}{\|\boldsymbol{z}_{\boldsymbol{e}}\| \|\boldsymbol{z}_{\boldsymbol{e}'}\|}.$$

Using this similarity function, the complete-linkage clustering algorithm[43] (Defays 1977) is used to extract relations classes. Since each pair end up in a single cluster, this assumes $\mathscr{H}_{\text{1-ADJACENCY}}$. Hasegawa et al. (2004) evaluate their method on articles from the New York Times (NYT). They extract relations classes by first clustering all $\boldsymbol{z}_{e_1,e_2}$ where e_1 has the type person and e_2 has the type GPE, and then by clustering all $\boldsymbol{z}_{e_1,e_2}$ where both e_1 and e_2 are organizations. By clustering separately different type combinations, they ensure that $\mathscr{H}_{\text{TYPE}}$ is enforced.

[43] The complete-linkage algorithm is an agglomerative hierarchical clustering method also called farthest neighbor clustering. The algorithm starts with each sample in its own cluster then merges the clusters two by two until reaching the desired number of clusters. At each step, the two closest clusters are merged together, with the distance between clusters being defined as the distance between their farthest elements.

They furthermore experiment with automatic labeling of the clusters with the most frequent word appearing in the samples. Apart from the relation *prime minister*, which is simply labeled "minister" since only unigrams are considered, the labels are rather on point. To measure the performance of their model, they use a classical supervised F_1 where each cluster is labeled by the majority gold relation. Using this somewhat unadapted metric, they reach an F_1 of 82% on person–GPE pairs and an F_1 of 77% on organization–organization pairs. This relatively high score compared to subsequent models can be explained by the small size of their dataset, which is further split by entity type. Furthermore, note that some generic relations such as *part of* do not follow $\mathscr{H}_{\text{TYPE}}$ and, as such, cannot be captured.

2.5.4 Rel-LDA

Rel-LDA (Yao et al. 2011) is a probabilistic generative model inspired by LDA. It works by clustering sentences: each relation defines a distribution over a handcrafted set of sentence features (Section 2.3.4) describing the relationship between the two entities in the text. Furthermore, rel-LDA models the propensity of a relation at the level of the document; thus, it is not strictly speaking a sentence-level relation extractor. The idea behind modeling this additional information is that when a relation such as P413 *position played on team* appears in a document, other relations pertaining to sports are more likely to appear. Figure 2.14 gives the plate diagram for the rel-LDA model. It uses the following variables:

Yao et al., "Structured Relation Discovery using Generative Models" EMNLP 2011

- $\mathbf{f}_i$ the features of the i-th sample, where f_{ij} is its j-th feature
- r_i the relation of the i-th sample
- θ_d the distribution of relations in the document d
- ϕ_{rj} the probability of the j-th feature to occurs for the relation r
- α the Dirichlet prior for θ_d
- β the Dirichlet prior for ϕ_{rj}

The generative process is listed as Algorithm 2.4. The learning process uses the expectation–maximization algorithm. In the variational E-step, the relation for each sample r_i is sampled from the categorical distribution:

$$P(r_i \mid \boldsymbol{f}_i, d) \propto P(r_i \mid d) \prod_{j=1}^{m} P(f_{ij} \mid r_i)$$

where $P(r \mid d)$ is defined by θ_d and $P(f_{ij} \mid r)$ is defined by ϕ_{rj}. In the M-step, the values for θ_d are computed by counting the number of times each relation appears in d and the hyperprior α; and the value for ϕ_{rj} is computed from the number of co-occurrences of the j-th feature with the relation r and from β.

Yao et al. (2011) evaluate their model on the New York Times by comparing their clusters to relations in Freebase. However, because of the incompleteness of knowledge bases, they only evaluate the recall on Freebase and use manual annotation to estimate the precision. Even though the original article lacks a significant comparison, subsequent approaches often compare to rel-LDA.

A first limitation of their approach is that given the relation r, the features f are independents. Since the entities are among those features, this means that $P(e_2 \mid e_1, r) = P(e_2 \mid r)$ which is clearly false.

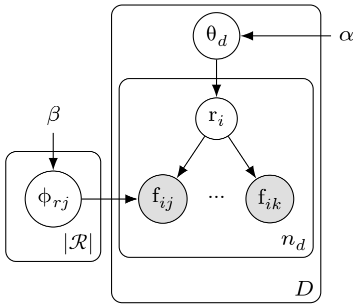


Figure 2.14: Rel-LDA plate diagram. D is the number of documents in the dataset and n_d is the number of samples in the document d. For each sample i, there are several features $f_{i1}, f_{i2}, \dots, f_{im}$, accordingly for each relation r, there are also several feature priors $\phi_{r1}, \dots, \phi_{rm}$, however for simplicity, a single prior is shown here.

algorithm Rel-LDA Generation
Inputs: α relations hyperprior
β features hyperprior
Output: $\boldsymbol{F}$ observed features

for all relations r **do**
  for all features j **do**
    Choose $\phi_{rj} \sim \text{Dir}(\beta)$
for all documents d **do**
  Choose $\theta_d \sim \text{Dir}(\alpha)$
  for all samples i in d **do**
    Choose $r \sim \text{Cat}(\theta_d)$
    for all features j **do**
      Choose $f_{ij} \sim \text{Cat}(\phi_{rj})$
output $\boldsymbol{F}$

Algorithm 2.4: The rel-LDA generative process. Dir are Dirichlet distributions. Cat are categorical distributions.

Assumption $\mathscr{H}_{\text{BICLIQUE}}$: *Given a relation, the entities are independent of one another:* $\mathrm{e}_1 \perp\!\!\!\perp \mathrm{e}_2 \mid \mathrm{r}$. *In other words, given a relation, all possible head entities are connected to all possible tail entities.*
$\forall r \in \mathcal{R} : \exists A, B \subseteq \mathcal{E} : r \bullet \breve{r} = \mathbf{1}_A \land \breve{r} \bullet r = \mathbf{1}_B$

This is a widespread problem with generative models which are inclined to make extensive independence assumptions. Furthermore, generative models have an implicit bias that all observed features are related to relation extraction, even though they might measures other aspect of the sample (style, idiolectal word choice, etc). This might results in the model focusing on features not related to the relation extraction task.

Several extensions of rel-LDA were proposed. Type-LDA (Yao et al. 2011) purpose to model entity types which are latent variables of entity features, themselves generated from the relation variable r, thus softly enforcing $\mathscr{H}_{\text{TYPE}}$. Sense-LDA (Yao et al. 2012) use a LDA-like model for each different dependency path. Clusters for different paths are then merged into relation clusters.

Yao et al., "Unsupervised Relation Discovery with Sense Disambiguation" ACL 2012

Rel-LDA is an important work in that it proposes a simple evaluation framework; in particular, it introduces the B^3 metric to unsupervised relation extraction. However, it predates the advent of neural networks and distributed representations in relation extraction, by which it was bound to be replaced.

2.5.5 Variational Autoencoder for Relation Extraction

Marcheggiani and Titov (2016) were first to propose a discriminative unsupervised relation extraction model. Discriminative models directly solve the inference problem of finding the posterior $P(r \mid x)$. This is in contrast to generative models such as rel-LDA which determine $P(x \mid r)$ and then use Bayes' theorem to compute $P(r \mid x)$ and make a prediction. The model of Marcheggiani and Titov (2016) is closely related to the approach presented in Chapter 3. It is a clustering model, meaning that it produces

Marcheggiani and Titov, "Discrete-State Variational Autoencoders for Joint Discovery and Factorization of Relations" TACL 2016 clusters of samples where the samples in each cluster all convey the same relation. To do so, it uses a variational autoencoder model (VAE, Kingma and Welling 2014) that we now describe.

Kingma and Welling, "Auto-Encoding Variational Bayes" ICLR 2014

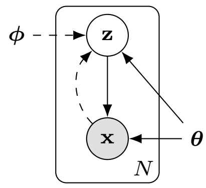


Figure 2.15: VAE plate diagram. N is the number of samples in the dataset.

Variational Autoencoder The goal of a variational autoencoder is to learn a latent variable $\boldsymbol{z}$ which explains the distribution of an observed variable $\boldsymbol{x}$. For our problem, the latent variable corresponds to the relation conveyed by the sample $\boldsymbol{x}$. We assume we know the generative process $P(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\theta})$, i.e. this process is the "decoder" (parametrized by $\boldsymbol{\theta}$): given the latent variable it produces a sample. However, the process of interest to us is to estimate the latent variable—the relation—from a sample, that is $P(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$. Using Bayes' theorem we can reformulate this posterior as $P(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\theta}) P(\boldsymbol{z} \mid \boldsymbol{\theta}) \mathbin{/} P(\boldsymbol{x} \mid \boldsymbol{\theta})$. However, computing $P(\boldsymbol{x} \mid \boldsymbol{\theta})$ is often intractable, especially when the likelihood $P(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\theta})$ is modeled using a complicated function like a neural network. To solve this problem, a variational approach is used: another model Q parametrized by $\boldsymbol{\phi}$ is used to approximate $P(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$ as well as possible. This approximation $Q(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\phi})$ is the "encoder" since it finds the latent variable associated with a sample. The model can then be trained by maximizing the log-likelihood given the latent variable estimated by Q and by minimizing the difference between the latent variable predicted by Q and the desired prior $P(\boldsymbol{z} \mid \boldsymbol{\theta})$:

$$J_{\text{ELBO}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathop{\mathbb{E}}_{Q(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\phi})} [\log P(\boldsymbol{x} \mid \boldsymbol{z}; \boldsymbol{\theta})] - \mathrm{D}_{\text{KL}}(Q(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\phi}) \parallel P(\boldsymbol{z} \mid \boldsymbol{\theta})) \quad (2.8)$$

A justification for this objective can also be found in the fact that it's a lower bound of the log marginal likelihood $\log P(\boldsymbol{x} \mid \boldsymbol{\theta})$, hence its name: evidence lower bound (ELBO). The first part of the objective is often referred to as the negative reconstruction loss since it seeks to reconstruct the sample $\boldsymbol{x}$ after it went through the encoder Q and the decoder P. One last problem with the VAE approximation relates to the reconstruction loss, the estimation of the expectation over $Q(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\phi})$ not being differentiable which makes the model—in particular $\boldsymbol{\phi}$—untrainable by gradient descent. This is usually solved using the reparameterization trick: sampling from $Q(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\phi})$ can often be done in a two steps process: sampling from a simple distribution like $\epsilon \sim \mathcal{N}(0, 1)$ then transforming this sample using a deterministic process parametrized by $\boldsymbol{\phi}$. The plate diagram of the VAE is given Figure 2.15 where the model P is marked with solid lines and the variational approximation Q is marked with dashed lines.

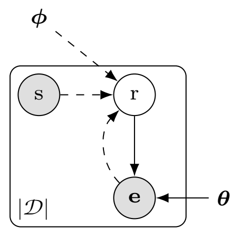


Figure 2.16: Marcheggiani and Titov (2016) plate diagram.

Coming back to the model of Marcheggiani and Titov (2016), it is a conditional β-VAE,[44] i.e. the whole process is conditioned on an additional variable. Indeed, in their approach, only the entities $\boldsymbol{e} \in \mathcal{E}^2$ are reconstructed, while the sentence $s \in \mathcal{S}$ simply conditions the whole process. The latent variable explaining the observed entities is expected to be the relation conveyed by the sample. The resulting model's plate diagram is given in Figure 2.16. This approach is defined by two models:

[44] The β in "β-VAE" simply indicates that the Kullback–Leibler term in Equation 2.8 is weighted by a hyperparameter β. More details are given in Chapter 3.

The Encoder $Q(\mathrm{r} \mid \boldsymbol{e}, s; \boldsymbol{\phi})$ is the relation extraction model properly speaking. It is defined as a linear model on top of handcrafted features (Section 2.3.4). For each sample, the model outputs a distribution over a predefined number of relations.

The Decoder $P(\boldsymbol{e} \mid r; \boldsymbol{\theta})$ is a model estimating how likely it is for two entities to be linked by a relation. It is a reconstruction model since the entities $\boldsymbol{e}$ are known and need to be retrieved from the latent relation r sampled from the encoder. It is defined using selectional preferences (Section 1.4.2.1) and RESCAL (Section 1.4.2.2).

Note that to label a sample $(\boldsymbol{e}, s) \in \mathcal{D}$, Marcheggiani and Titov (2016) simply select $\operatorname{argmax}_{r\in\mathcal{R}} Q(r \mid \boldsymbol{e}, s; \boldsymbol{\phi})$, meaning that the decoder is not used during evaluation. Its sole purpose is to provide a supervision signal to the encoder through the maximization of J_{ELBO}. The whole autoencoder can also be interpreted as being trained by a surrogate task of filling-in entity blanks. This is the interpretation we use in Chapter 3.

For Equation 2.8 to be well defined, a prior on the relations must also be selected; Marcheggiani and Titov (2016) make the following assumption:

Assumption $\mathscr{H}_{\text{UNIFORM}}$: *All relations occur with equal frequency.*
$\forall r \in \mathcal{R}\colon P(r) = \dfrac{1}{|\mathcal{R}|}$

They evaluate their approach on the New York Times distantly supervised by Freebase. By inducing 100 clusters, they show an improvement of the B^3 F_1 compared to DIRT (Section 2.3.3) and rel-LDA (Section 2.5.4). They also experiment using semi-supervised evaluation (Section 2.5.1) by pre-training their decoder on a subset of Freebase before training their encoder as described above; this additional supervision improves the F_1 by more than 27%. These results were further improved by Yuan and Eldardiry (2021), which proposed to split the latent variable into a relation r and sentence information z, with z conditioned on r and using a loss including the reconstruction of the sentence s from z.

2.5.6 Matching the Blanks

Soares et al., "Matching the Blanks: Distributional Similarity for Relation Learning" ACL 2019

Matching the blanks (MTB, Soares et al. 2019) is an unsupervised method that does not attempt to cluster samples but rather learns a representation of the relational semantics they convey. More precisely, this representation is used to measure the similarity between samples such that similar samples convey similar relations. As such, it is either evaluated as a supervised pre-training method (Section 2.5.1) or using a few-shot dataset (Section 2.5.1.2). The MTB article introduces several methods to extract an entity-aware representation of a sentence using BERT; this was discussed in Section 2.3.7. This section focuses on the unsupervised training. As a reminder, we refer to sentence encoder of MTB by the function $\text{BERTcoder}\colon \mathcal{S} \to \mathbb{R}^d$ illustrated Figure 2.7. Given this encoder, MTB defines the similarity between samples as:

$$\operatorname{sim}(s, s') = \sigma(\text{BERTcoder}(s)^\mathsf{T}\, \text{BERTcoder}(s')) \tag{2.9}$$

This similarity function can be used to evaluate the model on a few-shot task. Note that this function completely ignores entities identifiers (e.g. `Q211539`), but can still exploit the entities surface forms (e.g. "Peter Singer") through the sentence $s \in \mathcal{S}$. This model can be used as is, without any training other than the masked language model pre-training of BERT (Section 1.3.4.2) and reach an accuracy of 72.9% on the FewRel 5 way 1 shot dataset.

Soares et al. (2019) propose a training objective to fine-tune BERT for the unsupervised relation extraction task. This objective is called matching the blanks. It assumes that two sentences containing the same entities convey the same relation. This is exactly $\mathscr{H}_{\text{1-ADJACENCY}}$ as given Section 2.3.2. The probability that two sentences convey the same relation (D = 1) is taken from the similarity function: $P(\mathrm{D}=1 \mid s, s') = \operatorname{sim}(s, s')$. Given this, the $\mathscr{H}_{\text{1-ADJACENCY}}$ assumption is translated into the following negative sampling (Section 1.2.1.3) loss:

$$\mathcal{L}_{\text{MTB}} = \frac{-1}{|\mathcal{D}|^2} \sum_{\substack{(\boldsymbol{e},s)\in\mathcal{D} \\ (\boldsymbol{e}',s')\in\mathcal{D}}} \begin{aligned}[t] &\delta_{\boldsymbol{e},\boldsymbol{e}'} \log P(\mathrm{D}=1 \mid s, s') \\ &+(1-\delta_{\boldsymbol{e},\boldsymbol{e}'}) \log P(\mathrm{D}=0 \mid s, s') \end{aligned} \tag{2.10}$$

This loss is minimized through gradient descent by sampling random positive and negative sentence pairs. These pairs can be obtained by comparing the entity identifier without the need for any supervision.

A problem with this approach is that the BERTcoder model can simply learn to perform entity linking on the entities surface forms in the sentences s, thus minimizing Equation 2.10 by predicting whether $\boldsymbol{e} = \boldsymbol{e}'$. We want to avoid this since this would only work on samples seen during training and would not generalize to unseen entities. To ensure the model predicts whether the samples convey the same relation from the sentences s and s' alone, blanks are introduced. A special token <BLANK/> is substituted to the entities as follow:

> <BLANK/>$_{e_1}$, inspired by Cale's earlier cover, recorded one of the most acclaimed versions of "<BLANK/>$_{e_2}$."
> <BLANK/>$_{e_1}$'s rendition of "<BLANK/>$_{e_2}$" has been called "one of the great songs" by Time…

This is similar to the sample corruption of BERT (Section 1.3.4.2), indeed like BERT, the entity surface forms are blanked only a fraction[45] of the time so as to not confuse the model when real entities appear during evaluation.

[45] Soares et al. (2019) blanks each entity with a probability of 70%, meaning that only 9% of training samples have both of their entity surface forms intact.

Another problem with Equation 2.10 is that the negative sample space $\boldsymbol{e} \neq \boldsymbol{e}'$ is extremely large. Instead of taking negative samples randomly in this space, Soares et al. (2019) propose to take only samples which are likely to be close to positive ones. To this end, the $\boldsymbol{e} \neq \boldsymbol{e}'$ condition is actually replaced with the following one:

$$|\{e_1, e_2\} \cap \{e'_1, e'_2\}| = 1$$

These are called "strong negatives": negative samples that have precisely one entity in common. Negative sampling, especially with strong negatives, leads to another unfortunate assumption:

Assumption $\mathscr{H}_{1\to 1}$: *All relations are one-to-one.*
$\forall r \in \mathcal{R}\colon r \bullet \breve{r} \cup \boldsymbol{I} = \breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$

Indeed, if a relation is not one-to-one, then there exists two facts $e_1\ r\ e_2$ and $e_1\ r\ e_3$ (or respectively with $\breve{r}$); however these two facts form a strong negative pair, therefore as per $\mathcal{L}_{\text{MTB}}$ their representations must be pulled away from one another.

Despite these assumptions, MTB showcase impressive results, both as a few-shot and supervised pre-training method. It obtained state-of-the-art results both on the SemEval 2010 Task 8 dataset with a macro-$\overleftarrow{F_1}$ of 82.7% and on FewRel with an accuracy of 90.1% on the 5 way 1 shot task.

As a reminder, $\overleftarrow{F_1}$ is the half-directed metric described Section 2.3.1. It is referred to as "taking directionality into account" in the SemEval dataset.

2.5.7 SelfORE

SelfORE (X. Hu et al. 2020) is a clustering approach similar to the one of Hasegawa et al. (2004) presented in Section 2.5.3 but using deep neural network models for extracting sentence representations and for grouping these representations into relation clusters. Since they follow the experimental setup of Simon et al. (2019), which we present in Chapter 3, their results are listed in that chapter.

X. Hu et al., "SelfORE: Self-supervised Relational Feature Learning for Open Relation Extraction" EMNLP 2020

SelfORE uses MTB's entity markers–entity start BERTcoder sentence representation. A clustering algorithm could be run to produce relation classes from these representations a la Hasegawa et al. (2004). However, X. Hu et al. (2020) introduce an iterative scheme to purify the clusters. This scheme is illustrated in Figure 2.17 and works by alternatively optimizing two losses $\mathcal{L}_{\text{AC}}$ and $\mathcal{L}_{\text{RC}}$.

The first loss $\mathcal{L}_{\text{AC}}$ is the clustering loss which comes from DEC (Xie et al. 2016). DEC is a deep clustering algorithm that uses a denoising autoencoder (Vincent et al. 2010) to compress the input. In their case, the input $\boldsymbol{h}$ is the sentence encoded by BERTcoder. The denoising autoencoder is trained layer by layer with a small bottleneck which produces a compressed representation of the sentence $\boldsymbol{z} = \text{Encoder}(\boldsymbol{h})$. This is the space in which the clustering occurs. For each cluster $j = 1, \dots, K$, a centroid[46] $\boldsymbol{\mu}_j$ is learned such that a sentence is part of the cluster whose centroid is the closest to its compressed representation. This is modeled with a Student's t-distribution with one degree of freedom centered around the centroid:

Xie et al., "Unsupervised Deep Embedding for Clustering Analysis" ICML 2016

$$q_{ij} = \frac{(1 + \|\boldsymbol{z}_i - \boldsymbol{\mu}_j\|^2)^{-1}}{\sum_k (1 + \|\boldsymbol{z}_i - \boldsymbol{\mu}_k\|^2)^{-1}}$$

To force the initial clusters to be more distinct, a target distribution p is defined as:

$$p_{ij} = \frac{q_{ij}^2 \,/\, f_j}{\sum_k q_{ik}^2 \,/\, f_k} \tag{2.11}$$

where $f_j = \sum_i q_{ij}$ are soft cluster frequencies. To push $\boldsymbol{Q}$ towards $\boldsymbol{P}$, a Kullback–Leibler divergence is used:

$$\mathcal{L}_{\text{AC}} = \mathrm{D}_{\text{KL}}(\boldsymbol{P} \parallel \boldsymbol{Q}) = \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{K} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

This loss is minimized by backpropagating to the cluster centroids $\boldsymbol{\mu}_j$ and to the encoder's parameters in the DAE. Note that the decoder of the DAE is only used for initializing the encoder such that the input can be reconstructed.

[46] The k-means clustering algorithm is used to initialize the centroids. In practice, the k-means clusters could directly be used as soft labels. However, X. Hu et al. (2020) show that this underperforms compared to refining the clusters with $\mathcal{L}_{\text{AC}}$.

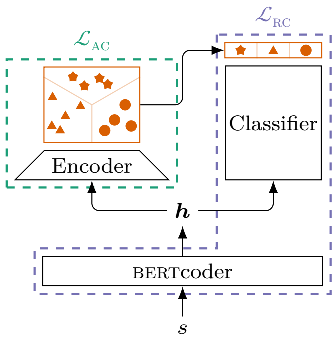


Figure 2.17: SelfORE iterative algorithm.

Optimizing $\mathcal{L}_{\text{AC}}$ is the first step of SelfORE; it assigns a pseudo-label to each sample in the dataset. The second step is to train a classifier to predict these pseudo-labels. The classifier is a simple multi-layer perceptron trained with the usual cross-entropy classification loss, which is called $\mathcal{L}_{\text{RC}}$ in SelfORE. This loss also backpropagate to the BERTcoder thus changing the sentence representations $\boldsymbol{h}$. SelfORE is an iterative algorithm: changing the $\boldsymbol{h}$ modifies the clustering found by DEC. Thus, the two steps, clustering and classification, are repeated several times until a stable label assignment is found.

The central assumption of SelfORE is that BERTcoder already produces a good representation for relation extraction, which, as we saw with the non-fine-tuned BERTcoder score on FewRel in Section 2.5.6, is rather accurate. However, SelfORE also assumes $\mathscr{H}_{\text{UNIFORM}}$, i.e. that all relations appear with the same frequency. This assumption is enforced by $\mathcal{L}_{\text{AC}}$, through the normalization of the target distribution $\boldsymbol{P}$ by soft cluster frequencies f_j.[47] Indeed, the distribution $\boldsymbol{P}$ is the original distribution $\boldsymbol{Q}$ more concentrated (because of the square) and more uniform (because of the normalization by f_j).

The interpretation of the concentration effect in terms of modeling hypotheses is more complex. The variable $\boldsymbol{h}$ is the concatenation of the two entity embeddings. Let's break down the BERTcoder function into two components: $\text{ctx}_1(s)$ and $\text{ctx}_2(s)$. These are simply the two contextualized embeddings of `<e1>` and `<e2>` (Section 2.5.6), in other words the function ctx contextualize an entity surface form inside its sentence. When two sentence representations $\boldsymbol{h}$ and $\boldsymbol{h}'$ are close, their pseudo-labels tend to be the same, and thus their relation also tend to be the same. In other words:

Assumption $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$: *Two samples with the same contextualized representation of their entities' surface forms convey the same relation.*
$\forall (s, \boldsymbol{e}, r), (s', \boldsymbol{e}', r') \in \mathcal{D}_{\mathcal{R}}:$

$$\text{ctx}_1(s) = \text{ctx}_1(s') \land \text{ctx}_2(s) = \text{ctx}_2(s') \implies r = r'$$

If we assume BERTcoder only performs entity linking of the entities surface form, then $\text{ctx}_i(s) = e_i$ for $i = 1, 2$, in this case $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$ collapses to $\mathscr{H}_{\text{1-ADJACENCY}}$, the contextualization inside the sentence s is ignored. On the other hand, if we assume BERTcoder provides no information about the entities and only encode the sentence, then $\text{ctx}_i(s) = s$ for $i = 1, 2$ and $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$ only states that the entity identifiers $\boldsymbol{e} \in \mathcal{E}^2$ should have no influence on the relation. The effective repercusion of $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$ lies somewhere half-way between these two extremes.

2.6 Conclusion

In this chapter, we introduced the relation extraction tasks (Section 2.1) and the different supervision schema with which we can tackle them (Section 2.2). As we showed, the development of supervised relation extraction models closely followed the evolution of NLP models introduced in Section 1.3. This is particularly visible in Section 2.3, which follows the progress of sentential relation extraction approaches. Furthermore, the expansion of the scale at which problems are tackled is visible both on the NLP side with the word-level to sentence-level evolution and on the information extraction side with the sentential to aggregate extraction evolution. The aggregate models, which are more aligned with the information extraction field, are presented in Section 2.4. Within these models, we also see the evolution from the simple max-pooling of MIML (Section 2.4.2) toward more sophisticated approaches which model the topology of the dataset more finely (Section 2.4.5).

We limited our presentation of supervised models to those critical to the development of unsupervised models. Several recent approaches propose to reframe supervised relation extraction—and other tasks—as language modeling (Raffel et al. 2020) or question answering (Cohen et al. 2021) tasks. Since these approaches were not explored in the unsupervised setup yet, we omit them from our related work.

[47] For further details, Xie et al. (2016) contains an analysis of the DEC clustering algorithm on imbalanced MNIST data.

Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer" JMLR 2020

Cohen et al., "Relation Classification as Two-way Span-Prediction" *under review* 2021

Finally, Section 2.5 focused on the specific setup of interest to this thesis: unsupervised relation extraction. This setup is particularly complex due to the discrepancy between the expressiveness of our supervised models and the weakness of the semantic signal we are seeking to extract. As we saw, modeling hypotheses are central to tackling this problem. Early models, including supervised ones, relied on strong hypotheses to facilitate training. However, while supervised models can now use deep neural networks without any hypothesis other than the unbiasedness of their data, unsupervised models still need to rely on strong assumptions.

In the next section, we focus on unsupervised discriminative models, in particular the VAE model presented in Section 2.5.5. In particular, we propose better losses for enforcing $\mathscr{H}_{\text{UNIFORM}}$, which avoid problematic degenerate solutions of the clustering relation extraction task.

Chapter 3

Regularizing Discriminative Unsupervised Relation Extraction Models

> *And once again I am I will not say alone, no, that's not like me, but, how shall I say, I don't know, restored to myself, no, I never left myself, free, yes, I don't know what that means but it's the word I mean to use, free to do what, to do nothing, to know, but what, the laws of the mind perhaps, of my mind, that for example water rises in proportion as it drowns you and that you would do better, at least no worse, to obliterate texts than to blacken margins, to fill in the holes of words till all is blank and flat and the whole ghastly business looks like what is, senseless, speechless, issueless misery.*
>
> — Samuel Beckett, *Molloy* (1955)

> *Careful! We don't want to learn anything from this.*
>
> — Bill Watterson, *Calvin and Hobbes* (1992)

This chapter is an adaptation of an article published at ACL with some supplementary results:
Étienne Simon et al. (July 2019). "Unsupervised Information Extraction: Regularizing Discriminative Approaches with Relation Distribution Losses". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 1378–1387. DOI: 10.18653/v1/P19-1133. URL: https://www.aclweb.org/anthology/P19-1133

All the works presented thus far follow the same underlying dynamic. There is a movement away from symbolic representations toward distributed ones, as well as a movement away from shallow models toward deeper ones. This can be seen in word, sentence and knowledge base representations (Chapter 1), as well as in relation extraction (Chapter 2). As we exposed in Chapter 2, a considerable amount of work has been conducted on supervised or weakly-supervised relation extraction (Sections 2.3 and 2.4), with recent state-of-the-art models using deep neural networks (Section 2.3.6). However, human annotation of text with knowledge base triplets is expensive and virtually impractical when the number of relations is large. Weakly-supervised methods such as distant supervision (Section 2.2.2) are also restricted to a handcrafted relation domain. Going further, purely unsupervised relation extraction methods working on raw texts, without any access to a knowledge base, have been developed (Section 2.5).

The first unsupervised models used a clustering (Section 2.5.3) or generative (Section 2.5.4) approach. The latter, which obtained state-of-the-art performance, still makes a lot of simplifying hypotheses, such as $\mathscr{H}_{\text{BICLIQUE}}$, assuming that the entities are conditionally independent between themselves given the relation. We posit that discriminative approaches can help further expressiveness, especially considering recent results with neural network models. The open question then becomes how to provide a sufficient learning signal to the classifier. The VAE model of Marcheggiani and Titov (2016) introduced in Section 2.5.5 followed this path by leveraging representation learning for modeling knowledge bases and proposed to use an auto-encoder model: their encoder extracts the relation from a sentence that the decoder uses to predict a missing entity. However, their encoder is still limited compared to its supervised counterpart (e.g. PCNN) and relies on handcrafted features extracted by natural language processing tools (Section 2.3.4). These features tend to contain errors and prevent the discovery of new patterns, which might hinder performances.

While the transition to deep learning approaches can bring more expressive models to the task, it also raises new problems. This chapter tackles a problem specific to unsupervised discriminative relation extraction models. In particular, we focus on the VAE model of Section 2.5.5. These models tend to be hard to train because of the way $\mathscr{H}_{\text{UNIFORM}}$ is enforced, expressly, how we ensure that all relations are conveyed the same amount of time.[48] To tackle this issue, we propose two new regularizing losses on the distribution of relations. With these, we hope to leverage the expressivity of discriminative approaches—in particular, of deep neural network classifiers—while staying in an unsupervised setting. Indeed, these models are hard to train without supervision, and the solutions proposed at the time were unstable. Discriminative approaches have less inductive bias, but this makes them more sensitive to noise.

[48] However, this problem can be generalized to how we enforce all relations are conveyed reasonably often.

Indeed, our initial experiments showed that the VAE relation extraction model was unstable, especially when using a deep neural network relation classifier. It converges to either of the two following regimes, depending on hyperparameter settings: always predicting the same relation or predicting a uniform distribution. To overcome these limitations, we propose to use two new losses alongside an entity prediction loss based on a fill-in-the-blank task and show experimentally that this is key to learning deep neural network models. Our contributions are the following:

- We propose two RelDist losses: a skewness loss, which encourages the classifier to predict a class with confidence for a single sentence, and a distribution distance loss, which encourages the classifier to scatter a set of sentences into different classes;
- We perform extensive experiments on the usual NYT + FB dataset, as well as two new datasets;
- We show that our RelDist losses allow us to train a deep PCNN classifier and improve the performances of feature-based models.

In this chapter, we first describe our model in Section 3.1 before revisiting the related works pertinent to the experimental setup in Section 3.2. We present our main experimental results in Section 3.3 before studying some possible improvements we considered in Section 3.4.

3.1 Model description

Our model focuses on extracting the relation between two entities in textual data and assumes that an entity chunker has identified named entities in the text. Furthermore, following Section 2.1, we limit ourselves to binary relations and therefore consider sentences with two tagged entities, as shown in Figure 3.1. These sentences constitute the set $\mathcal{S}$. We further assume that entity linking was performed and that we have access to entity identifiers from the set $\mathcal{E}$. We therefore consider samples from a dataset $\mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}^2$. From these samples we learn a relation classifier that maps each sample $x \in \mathcal{D}$ to a relation $r \in \mathcal{R}$. As such, our approach is sentential (Section 2.1).

To provide a supervision signal to our relation classifier, we follow the VAE model of Section 2.5.5 (Marcheggiani and Titov 2016). However, the interpretation of their model as a VAE is part of the limitation we observed and is in conflict with the modifications we introduce. We, therefore, reformulate their approach as a *fill-in-the-blank* task:

Marcheggiani and Titov, "Discrete-State Variational Autoencoders for Joint Discovery and Factorization of Relations" TACL 2016

"The $\underline{\text{sol}}_{e_1}$ was the currency of $\underline{\ ?\ }_{e_2}$ between 1863 and 1985."

$$\underbrace{\text{The}}_{\text{prefix}}\ \overbrace{\textbf{sol}}^{\text{head entity}}\ \underbrace{\text{was the currency of}}_{\text{infix}}\ \overbrace{\textbf{Peru}}^{\text{tail entity}}\ \underbrace{\text{between 1863 and 1985.}}_{\text{suffix}}$$

Figure 3.1: A sentence from Wikipedia where the conveyed relation is "*currency used by*." In contrast to Figure 2.3, which presented DIPRE's split-in-three-affixes, we do not label the entities surface forms with e_1 and e_2 to avoid confusion with entity identifiers.

To correctly fill in the blank, we could directly learn to predict the missing entity; but in this case, we would not be able to learn a relation classifier. Instead, we first want to learn that this sentence expresses the semantic relation "currency used by" before using this information for a (self-)supervised entity prediction task. To this end, we make the following assumption:

Assumption $\mathscr{H}_{\text{BLANKABLE}}$: *The relation can be predicted by the text surrounding the two entities alone. Formally, using* blanked(s) *to designate the tagged sentence $s \in \mathcal{S}$ from which the entities surface forms were removed, we can write:*

$\mathrm{r} \perp\!\!\!\perp \mathbf{e} \mid \text{blanked(s)}$.

Furthermore, since the information between s and blanked(s) is determined by $\mathbf{e}$, as a corollary of $\mathscr{H}_{\text{BLANKABLE}}$, we have the equivalence $P(\mathrm{r} \mid \mathrm{s}) = P(\mathrm{r} \mid \text{blanked(s)})$. Using this assumption and the above observation about filling blanked entities, we design a surrogate fill-in-the-blank task to train a relation extraction model. This task uses the point of view that a relation is something that allows us to predict e_2 from e_1 and vice versa. Our goal is to predict a missing entity e_{-i} given the predicted relation r and the other entity e_i:

$$P(e_{-i} \mid s, e_i) = \sum_{r\in\mathcal{R}} \underbrace{P(r \mid s)}_{\text{(i) classifier}} \underbrace{P(e_{-i} \mid r, e_i)}_{\text{(ii) entity predictor}} \qquad \text{for } i = 1, 2, \tag{3.1}$$

Derivation of Equation 3.1:
$P(e_{-i} \mid s, e_i)$
First introduce and marginalize the latent relation variable r ("sum rule"):
$= \sum_{r\in\mathcal{R}} P(r, e_{-i} \mid s, e_i)$
Apply the definition of conditional probability ("product rule"):
$= \sum_{r\in\mathcal{R}} P(r \mid s, e_i) P(e_{-i} \mid r, s, e_i)$
Apply the independence $\mathscr{H}_{\text{BLANKABLE}}$ assumption on the first term and our definition of a relation on the second:
$= \sum_{r\in\mathcal{R}} P(r \mid s) P(e_{-i} \mid r, e_i)$
Furthermore, by applying the corollary of $\mathscr{H}_{\text{BLANKABLE}}$, we can write:
$= \sum_{r\in\mathcal{R}} P(r \mid \text{blanked}(s)) P(e_{-i} \mid r, e_i)$

where $e_1, e_2 \in \mathcal{E}$ are the two entities identifiers, $s \in \mathcal{S}$ is the sentence mentioning them, and $r \in \mathcal{R}$ is the relation linking them. As the entity predictor can consider either entity, we use e_i to designate the given entity, and $e_{-i} = \{e_1, e_2\} \setminus \{e_i\}$ the one to predict.

The relation classifier $P(r \mid s)$ and entity predictor $P(e_{-i} \mid r, e_i)$ are trained jointly to discover a missing entity, with the constraint that the entity predictor cannot access the input sentence directly. Thus, all the required information must be condensed into r, which acts as a bottleneck. We advocate that this information is the semantic relation between the two entities.

Note that Marcheggiani and Titov (2016) did not make the $\mathscr{H}_{\text{BLANKABLE}}$ hypothesis. Instead, their classifier is conditioned on both e_i and e_{-i}, strongly relying on the fact that r is an information bottleneck and will not "leak" the identity of e_{-i}. This is possible since they use pre-defined sentence representations; this is impossible to enforce with the learned representations of a deep neural network.

In the following, we first describe the relation classifier $P(r \mid s)$ in Section 3.1.1 before introducing the entity predictor $P(e_{-i} \mid r, e_i)$ in Section 3.1.2. Arguing that the resulting model is unstable, we describe the two new RelDist losses in Section 3.1.3.

3.1.1 Unsupervised Relation Classifier

Our model for $P(r \mid s)$ follows the then state-of-the-art practices for supervised relation extraction by using a piecewise convolutional neural network

(PCNN, Section 2.3.6, Zeng et al. 2015). Similar to DIPRE's split-in-three-affixes, the input sentence can be split into three parts separated by the two entities (see Figure 3.1). In a PCNN, the model outputs a representation for each part of the sentence. These are then combined to make a prediction. Figure 2.6 shows the network architecture that we now describe.

Zeng et al., "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks" EMNLP 2015

First, each word of s is mapped to a real-valued vector. In this work, we use standard word embeddings, initialized with GloVe[49] (Section 1.2.1, Pennington et al. 2014), and fine-tune them during training. Based on those embeddings, a convolutional layer detects patterns in subsequences of words. Then, a max-pooling along the text length combines all features into a fixed-size representation. Note that in our architecture, we obtained better results by using three distinct convolutions, one for each sentence part (i.e. the weights are not shared). We then apply a non-linear function (tanh) and sum the three vectors into a single representation for s. Finally, this representation is fed to a softmax layer to predict the distribution over the relations. This distribution can be plugged into Equation 3.1. Denoting PCNN our classifier, we have:

[49] We use the `6B.50d` pre-trained word embeddings from https://nlp.stanford.edu/projects/glove/

$$P(r \mid s) = \text{PCNN}(r; s, \boldsymbol{\phi}),$$

where $\boldsymbol{\phi}$ are the parameters of the classifier. Note that we can use the PCNN to predict the relationship for any pair of entities appearing in any sentence since the input will be different for each selected pair (see Figure 2.6). Furthermore, since the PCNN ignore the entities surface forms, we can have $P(r \mid s) = P(r \mid \text{blanked}(s))$ which is necessary to enforce $\mathscr{H}_{\text{BLANKABLE}}$.

3.1.2 Entity Predictor

The purpose of the entity predictor is to provide supervision for the relation classifier. As such, it needs to be differentiable. We follow Marcheggiani and Titov (2016) to model $P(e_i \mid r, e_{-i})$, and use an energy-based formalism, where $\psi(e_1, r, e_2)$ is the energy associated with (e_1, r, e_2). The probability is obtained as follows:

$$P(e_1 \mid r, e_2) = \frac{\exp(\psi(e_1, r, e_2))}{\sum_{e' \in \mathcal{E}} \exp(\psi(e', r, e_2))}, \tag{3.2}$$

where ψ is expressed as the sum of two standard relational learning models selectional preferences (Section 1.4.2.1) and RESCAL (Section 1.4.2.2):

$$\psi(e_1, r, e_2; \boldsymbol{\theta}) = \underbrace{\boldsymbol{u}_{e_1}^\top \boldsymbol{a}_r + \boldsymbol{u}_{e_2}^\top \boldsymbol{b}_r}_{\text{Selectional Preferences}} + \underbrace{\boldsymbol{u}_{e_1}^\top \boldsymbol{C}_r \boldsymbol{u}_{e_2}}_{\text{RESCAL}}$$

where $\boldsymbol{U} \in \mathbb{R}^{\mathcal{E} \times m}$ is an entity embedding matrix, $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{\mathcal{R} \times m}$ are two matrices encoding the preferences of each relation of certain entities, $\boldsymbol{C} \in \mathbb{R}^{\mathcal{R} \times m \times m}$ is a three-way tensor encoding the entities interactions, and the hyperparameter m is the dimension of the embedded entities. The function ψ also depends on the energy functions parameters $\boldsymbol{\theta} = \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{U}\}$ that we might omit for legibility. RESCAL (Nickel et al. 2011) uses a bilinear tensor product to gauge the compatibility of the two entities; whereas, in the Selectional Preferences model, only the predisposition of an entity to appear as the subject or object of a relation is captured.

Negative Sampling The number of entities being very large, the partition function of Equation 3.2 cannot be efficiently computed. To avoid the summation over the set of entities, we follow Section 1.2.1.3 and use negative sampling (Mikolov et al. 2013b); instead of training a softmax classifier, we train a discriminator which tries to recognize real triplets ($\mathrm{D}=1$) from fake ones ($\mathrm{D}=0$):

$$P(\mathrm{D}=1 \mid e_1, e_2, r) = \sigma\left(\psi(e_1, r, e_2)\right),$$

where $\sigma(x) = 1 \mathbin{/} (1+\exp(-x))$ is the sigmoid function. This model is then trained by generating negative entities for each position and optimizing the negative log-likelihood:

$$\begin{aligned}\mathcal{L}_{\text{EP}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathop{\mathbb{E}}_{\substack{(\mathrm{s},\mathrm{e}_1,\mathrm{e}_2)\sim\mathcal{U}(\mathcal{D})\\ \mathrm{r}\sim\textsc{pcnn}(\mathrm{s};\boldsymbol{\phi})}} \Bigg[& -\log\sigma\left(\psi(\mathrm{e}_1, \mathrm{r}, \mathrm{e}_2; \boldsymbol{\theta}) + b_{\mathrm{e}_1}\right) \\ & -\log\sigma\left(\psi(\mathrm{e}_1, \mathrm{r}, \mathrm{e}_2; \boldsymbol{\theta}) + b_{\mathrm{e}_2}\right) \\ & -\sum_{j=1}^{k} \mathop{\mathbb{E}}_{\mathrm{e}'\sim\mathcal{U}_{\mathcal{D}}(\mathcal{E})} \left[\log\sigma\left(-\psi(\mathrm{e}_1, \mathrm{r}, \mathrm{e}'; \boldsymbol{\theta}) - b_{\mathrm{e}'}\right)\right] \\ & -\sum_{j=1}^{k} \mathop{\mathbb{E}}_{\mathrm{e}'\sim\mathcal{U}_{\mathcal{D}}(\mathcal{E})} \left[\log\sigma\left(-\psi(\mathrm{e}', \mathrm{r}, \mathrm{e}_2; \boldsymbol{\theta}) - b_{\mathrm{e}'}\right)\right] \Bigg]\end{aligned} \tag{3.3}$$

This loss is defined over the empirical data distribution $\mathcal{U}(\mathcal{D})$, i.e. the samples $(\mathrm{s}, \mathrm{e}_1, \mathrm{e}_2)$ follow a uniform distribution over sentences tagged with two entities; and the empirical entity distribution $\mathcal{U}_{\mathcal{D}}(\mathcal{E})$, that is the categorical distribution over $\mathcal{E}$ where each entity is weighted by its frequency in $\mathcal{D}$. The distribution of the relation r for the sentence s is then given by the classifier $\textsc{pcnn}(\mathrm{s}; \boldsymbol{\phi})$, which corresponds to the $\sum_{r\in\mathcal{R}} P(r \mid s)$ in Equation 3.1. Following standard practice, during training, the expectation on negative entities is approximated by sampling k random entities following the empirical entity distribution $\mathcal{E}$ for each position.

Biases Following Marcheggiani and Titov (2016), we add a bias for entities to ψ. These biases are parametrized by a single vector $\boldsymbol{b} \in \mathbb{R}^{\mathcal{E}}$. They encode how some entities are more likely to appear than others; as such, the $+\boldsymbol{b}_{e_i}$ appear in $\mathcal{L}_{\text{EP}}$ where the $P(e_i \mid r, e_{-i})$ would appear in the negative sampling estimation.

Approximation When $|\mathcal{R}|$ is large, the expectation over $\mathrm{r} \sim \textsc{pcnn}(\mathrm{s}; \boldsymbol{\phi})$ can be slow to evaluate. To avoid computing ψ for all possible relation $r \in \mathcal{R}$, we employ an optimization also used by Marcheggiani and Titov (2016). This optimization is built upon the following approximation:

$$\mathop{\mathbb{E}}_{\mathrm{r}\sim\textsc{pcnn}(\mathrm{s};\boldsymbol{\phi})}\left[\log\sigma(\psi(\mathrm{e}_1, \mathrm{r}, \mathrm{e}_2; \boldsymbol{\theta}))\right] \approx \log\sigma\left(\mathop{\mathbb{E}}_{\mathrm{r}\sim\textsc{pcnn}(\mathrm{s};\boldsymbol{\phi})}\left[\psi(\mathrm{e}_1, \mathrm{r}, \mathrm{e}_2; \boldsymbol{\theta})\right]\right). \tag{3.4}$$

Since the function ψ is linear in r, we can efficiently compute its expected value over r using the convex combinations of the relation embeddings. For example we can replace the selectional preference of a relation r for a head entity e_1: $\boldsymbol{u}_{e_1}^{\mathsf{T}} \boldsymbol{a}_r$ by the selectional preference of a distribution $\textsc{pcnn}(s; \boldsymbol{\phi})$ for a head entity: $\boldsymbol{u}_{e_1}^{\mathsf{T}}(\textsc{pcnn}(s; \boldsymbol{\phi})^{\mathsf{T}} \boldsymbol{A})$.

3.1.3 RelDist losses

Training the classifier through Equation 3.3 alone is very unstable and dependent on precise hyperparameter tuning. More precisely, according to our early experiments, the training process usually collapses into one of two regimes:

($\mathscr{P}$1) The classifier is very uncertain about which relation is expressed and outputs a uniform distribution over relations (Figure 3.2);

($\mathscr{P}$2) All sentences are classified as conveying the same relation (Figure 3.3).

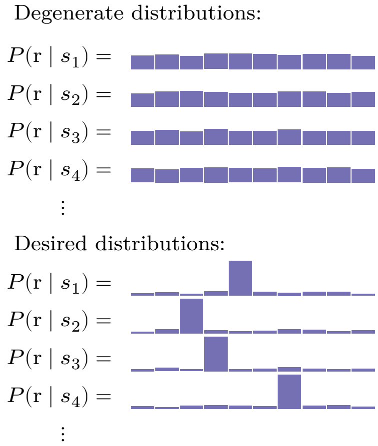


Figure 3.2: Illustration of $\mathscr{P}$ 1. The classifier assigns roughly the same probability to all relations. Instead, we would like the classifier to predict a single relation confidently.

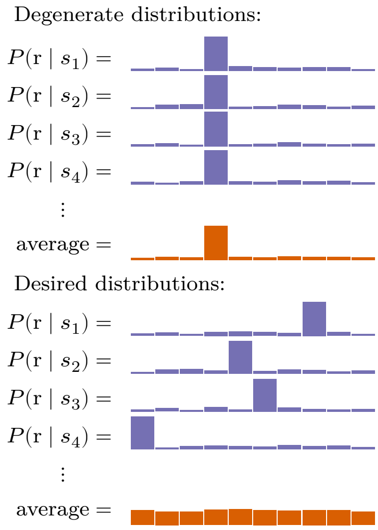


Figure 3.3: Illustration of $\mathscr{P}$ 2. The classifier consistently predicts the same relation. This is clearly visible when taking the average distribution (by marginalizing over the sentences s). Instead, we would like the classifier to predict a diverse set of relations.

In both cases, the entity predictor can do a good job minimizing $\mathcal{L}_{\text{EP}}$ by ignoring the output of the classifier, simply exploiting entities' co-occurrences. More precisely, many entities only appear in one relationship with a single other entity. In this case, the entity predictor can easily ignore the relationship r and predict the missing entity—and this pressure is even worse at the beginning of the optimization process as the classifier's output is not yet reliable.

This instability problem is particularly prevalent since the two components (classifier and entity predictor) are strongly interdependent: the classifier cannot be trained without a good entity predictor, which itself cannot take r into account without a good classifier resulting in a bootstrapping problem. To overcome these pitfalls, we developed two additional losses, which we now describe.

Skewness. Firstly, to encourage the classifier to be confident in its output, we minimize the entropy of the predicted relation distribution. This addresses $\mathscr{P}$1 by forcing the classifier toward outputting one-hot vectors for a given sentence using the following loss:

$$\mathcal{L}_{\text{S}}(\boldsymbol{\phi}) = \mathop{\mathbb{E}}_{(\text{s},\mathbf{e})\sim\mathcal{U}(\mathcal{D})} \left[\text{H}(\text{R} \mid \text{s}, \mathbf{e}; \boldsymbol{\phi})\right], \tag{3.5}$$

where R is the random variable corresponding to the predicted relation. Following our first independence hypothesis, the entropy of equation 3.5 is equivalent to $\text{H}(\text{R} \mid \text{s})$.

Distribution Distance. Secondly, to ensure that the classifier predicts several relations, we enforce $\mathscr{H}_{\text{UNIFORM}}$ by minimizing the Kullback–Leibler divergence between the model prior distribution over relations $P(\text{R} \mid \boldsymbol{\phi})$ and the uniform distribution[50] over the set of relations $\mathcal{U}(\mathcal{R})$, that is:

$$\mathcal{L}_{\text{D}}(\boldsymbol{\phi}) = \text{D}_{\text{KL}}(P(\text{R} \mid \boldsymbol{\phi}) \parallel \mathcal{U}(\mathcal{R})). \tag{3.6}$$

Note that contrary to $\mathcal{L}_{\text{S}}$, to have a good approximation of $P(\text{R} \mid \boldsymbol{\phi})$, the loss $\mathcal{L}_{\text{D}}$ measures the unconditional distribution over R, i.e. the distribution of predicted relations over all sentences. This addresses $\mathscr{P}$2 by forcing the classifier toward predicting each class equally often over a set of sentences.

To satisfactorily and jointly train the entity predictor and the classifier, we use the two losses at the same time, resulting in the final loss:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathcal{L}_{\text{EP}}(\boldsymbol{\theta}, \boldsymbol{\phi}) + \alpha\mathcal{L}_{\text{S}}(\boldsymbol{\phi}) + \beta\mathcal{L}_{\text{D}}(\boldsymbol{\phi}), \tag{3.7}$$

where α and β are both positive hyperparameters.

[50] Other distributions could be used, but in the absence of further information, this might be the best thing to do. See Section 3.5 for a discussion of alternatives.

All three losses are defined over the real data distribution, but in practice, they are approximated at the level of a mini-batch. First, both $\mathcal{L}_{\text{EP}}$ and $\mathcal{L}_{\text{S}}$ can be computed for each sample independently. To optimize $\mathcal{L}_{\text{D}}$ however, we need to estimate $P(\mathrm{R})$ at the mini-batch level and maximize the entropy of the mean predicted relation. Formally, let s_i for $i = 1, \dots, B$ be the i-th sentence in a batch of size B, we approximate $\mathcal{L}_{\text{D}}$ as:

$$\sum_{r \in \mathcal{R}} \left(\sum_{i=1}^{B} \frac{\text{PCNN}(r; s_i)}{B} \right) \log \left(\sum_{i=1}^{B} \frac{\text{PCNN}(r; s_i)}{B} \right).$$

Learning We optimize the empirical estimation of Equation 3.7, learning the PCNN parameters and word embeddings $\boldsymbol{\phi}$ as well as the entity predictor parameters and entity embeddings $\boldsymbol{\theta}$ jointly.

3.2 Related Work

The NLP and knowledge base related work is presented in Chapter 1, and the relation extraction related work is presented in Chapter 2. The main approaches we built upon are:

- Distant supervision (Section 2.2.2, Mintz et al. 2009): the method we use to obtain a supervised dataset for evaluation;[51]
- PCNN (Section 2.3.6, Zeng et al. 2015): our relation classifier, which was the state-of-the-art supervised relation extraction method at the time;
- Rel-LDA (Section 2.5.4, Yao et al. 2011): the state-of-the-art generative model we compare to;
- VAE for relation extraction (Section 2.5.5, Marcheggiani and Titov 2016): the overall inspiration for the architecture of our model, with which we share the entity predictor;
- SelfORE (Section 2.5.7, X. Hu et al. 2020): an extension of our work, which, alongside their own approach, proposed an improvement of our relation classifier by replacing the PCNN by a BERTcoder.

[51] As explained in Section 2.5.1.1, this is sadly standard in the evaluation of clustering approaches.

In this section, we give further details about the relationship between our losses and the ones derived by Marcheggiani and Titov (2016). As a reminder, their model is a VAE defined from an encoder $Q(r \mid \boldsymbol{e}, s; \boldsymbol{\phi})$ and a decoder $P(\boldsymbol{e} \mid r, s; \boldsymbol{\theta})$ as:

The prior of a conditional VAE $P(r \mid \boldsymbol{\theta})$ is usually conditioned on s too. However, this additional variable is not used by Marcheggiani and Titov (2016).

$$\mathcal{L}_{\text{VAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathop{\mathbb{E}}_{Q(r \mid \boldsymbol{e}, s; \boldsymbol{\phi})} [-\log P(\boldsymbol{e} \mid r, s; \boldsymbol{\theta})] + \beta \, \mathrm{D}_{\text{KL}}(Q(r \mid \boldsymbol{e}, s; \boldsymbol{\phi}) \parallel P(r \mid \boldsymbol{\theta})) \tag{3.8}$$

This is simply a rewriting of the ELBO of Equation 2.8 substituting relation extraction variables to the generic ones. There is however two differences compared to a standard VAE. First, the variable s is not reconstructed, it simply conditions the whole process. Second, the regularization term is weighted by a hyperparameter β. This makes the model of Marcheggiani and Titov (2016) a conditional β-VAE (Higgins et al. 2017; Sohn et al. 2015). The first summand of Equation 3.8 is called the reconstruction loss since it reconstructs the input variable $\boldsymbol{e}$ from the latent variable r and the conditional variable s. Since we followed the same structure for our

Higgins et al., "β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework" ICLR 2017
Sohn et al., "Learning Structured Output Representation using Deep Conditional Generative Models" NeurIPS 2015

model, this reconstruction loss is actually $\mathcal{L}_{\text{EP}}$, the difference being in the relation classifier. We can then rewrite the loss of Marcheggiani and Titov (2016) as:

As explained Section 2.5.5, Q is the VAE's encoder.

$$\mathcal{L}_{\text{VAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathcal{L}_{\text{EP}}(\boldsymbol{\theta}, \boldsymbol{\phi}) + \beta \mathcal{L}_{\text{VAE REG}}(\boldsymbol{\theta}, \boldsymbol{\phi})$$
$$\mathcal{L}_{\text{VAE REG}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathrm{D}_{\text{KL}}(Q(\mathrm{r} \mid \mathbf{e}; \boldsymbol{\phi}) \parallel P(\mathrm{r} \mid \boldsymbol{\theta}))$$

In their work, they select the prior as a uniform distribution over all relations $P(\mathrm{r} \mid \boldsymbol{\theta}) = \mathcal{U}(\mathcal{R})$ and approximate $\mathcal{L}_{\text{VAE REG}}$ as follow:

$$\mathcal{L}_{\text{VAE REG}}(\boldsymbol{\phi}) = \mathop{\mathbb{E}}_{(\mathrm{s}, \mathbf{e}) \sim \mathcal{U}(\mathcal{D})} [-\mathrm{H}(\mathrm{R} \mid \mathrm{s}, \mathbf{e}; \boldsymbol{\phi})]$$

Its purpose is to prevent the classifier from always predicting the same relation, i.e. it has the same purpose as our distance loss $\mathcal{L}_{\text{D}}$. However, its expression is equivalent to $-\mathcal{L}_{\text{S}}$, and indeed, minimizing the opposite of our skewness loss increases the entropy of the classifier output, addressing $\mathscr{P}2$ (classifier always outputting the same relation). Yet, using $\mathcal{L}_{\text{VAE REG}} = -\mathcal{L}_{\text{S}}$ alone, draws the classifier into the other pitfall $\mathscr{P}1$ (not predicting any relation confidently). In a traditional VAE, $\mathscr{P}1$ is addressed by the reconstruction loss $\mathcal{L}_{\text{EP}}$. However, at the beginning of training, the supervision signal is so weak that we cannot rely on $\mathcal{L}_{\text{EP}}$ for our task. The β weighting can be decreased to avoid $\mathscr{P}1$, but this would also lessen the solution to $\mathscr{P}2$. This causes a drop in performance, as we show experimentally.

3.3 Experiments

To compare with previous works, we repeat the experimental setup of Marcheggiani and Titov (2016) with the B^3 evaluation metric (Bagga and Baldwin 1998). We complemented this setup with two additional datasets extracted from T-REx (Elsahar et al. 2018) and two more metrics commonly seen in clustering task evaluation: V-measure (Rosenberg and Hirschberg 2007) and ARI (Hubert and Arabie 1985). This allows us to capture the characteristics of each approach in more detail.

In this section, we begin by describing the processing of the datasets in Section 3.3.1. We then describe the experimental details of the models we evaluated in Section 3.3.2. Finally, we give quantitative results in Section 3.3.3 and qualitative results in Section 3.3.4 The description of the metrics can be found in Section 2.5.1.1. Appendix C gives further details on the source datasets, their specificities, their sizes and some example of their content when appropriate.

3.3.1 Datasets

As explained in Section 2.5.1, to evaluate the models, we use labeled datasets, the labels being used for validation and testing. The first dataset we consider is the one of Marcheggiani and Titov (2016), which is similar to the one used in Yao et al. (2011). This dataset was built through distant supervision (Section 2.2.2) by aligning sentences from the New York Times corpus (NYT, Section C.5, Sandhaus 2008) with Freebase (FB, Section C.3, Bollacker et al. 2008) facts. Several sentences were filtered out based on features like the length of the dependency path between the two entities, resulting in 2 million sentences with only 41 000 (2%) of them labeled with one of 262 possible relations. 20% of the labeled sentences were set aside for validation; the remaining 80% are used to compute the final results.

We also extracted two datasets from T-REx (Section C.7, Elsahar et al. 2018), which was built as an alignment of Wikipedia with Wikidata (Section C.8, Vrandečić and Krötzsch 2014). We only consider (s, e_1, e_2) triplets where both entities appear in the same sentence.[52] If a single sentence contains multiple triplets, it appears multiple times in the dataset, each time with a different pair of tagged entities. We built the first dataset DS by extracting all triplets of T-REx where the two entities are linked by a relation in Wikidata. This is the usual distant supervision method. It results in 1 189 relations and nearly 12 million sentences, all of them labeled with a relation.

[52] T-REx provides annotations for whole articles; it should therefore be possible to process broader contexts by defining $\mathcal{S}$ as a set of articles. However, in this work, we stay in the traditional sentence-level relation extraction setup.

In Wikidata, each relation is annotated with a list of associated surface forms; for example, "*shares border with*" can be conveyed by "borders," "adjacent to," "next to," etc. The second dataset we built, SPO, only contains the sentences where a surface form of the relation also appears in the sentence, resulting in 763 000 samples (6% of the unfiltered dataset) and 615 relations. This dataset still contains some misalignment, but it should nevertheless be easier for models to extract the correct semantic relation since the set of surface forms is much more restricted and much more regular.

3.3.2 Baselines and Models

We compare our model with three state-of-the-art approaches, two generative rel-LDA models of Yao et al. (2011), the VAE model of Marcheggiani and Titov (2016) and the deep clustering of BERT representations by X. Hu et al. (2020).

The two rel-LDA models only differ by the number of features considered. We use the eight features listed in Marcheggiani and Titov (2016):

1. the bag of words of the infix;
2. the surface form of the entities;
3. the lemma words on the dependency path;
4. the POS of the infix words;
5. the type of the entity pair (e.g. person–location);
6. the type of the head entity (e.g. person);
7. the type of the tail entity (e.g. location);
8. the words on the dependency path between the two entities.

Rel-LDA uses the first three features, while rel-LDA1 is trained by iteratively adding more features until all eight are used.

To assess our two main contributions individually, we evaluate the PCNN classifier and our additional losses separately. More precisely, we first study the effect of the RelDist losses by looking at the differences between models optimizing $\mathcal{L}_{\text{EP}} + \mathcal{L}_{\text{VAE REG}}$ and the ones optimizing $\mathcal{L}_{\text{EP}} + \mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$ with $\mathcal{L}_{\text{EP}}$ being either computed using the relation classifier of Marcheggiani and Titov (2016) or our PCNN. Second, we study the effect of the relation classifier by comparing the feature-based classifier and the PCNN trained with the same losses. We also give results for our RelDist losses together with a BERTcoder classifier. This latter combination is evaluated by X. Hu et al. (2020) following our experimental setup. We thus focus mainly on four models:

- Linear + $\mathcal{L}_{\text{VAE REG}}$, which corresponds to the model of Marcheggiani and Titov (2016);
- Linear + $\mathcal{L}_{\text{S}}$ + $\mathcal{L}_{\text{D}}$, which uses the feature-based linear encoder of Marcheggiani and Titov (2016) together with our RelDist losses;
- PCNN + $\mathcal{L}_{\text{VAE REG}}$, which uses our PCNN encoder together with the regularization of Marcheggiani and Titov (2016);
- PCNN + $\mathcal{L}_{\text{S}}$ + $\mathcal{L}_{\text{D}}$, which is our complete model.

All models are trained with ten relation classes, which, while lower than the number of actual relations, allows us to compare the models faithfully since the distribution of gold relations is very unbalanced. For feature-based models, the size of the features domain range from 1 to 10 million values depending on the dataset. We train our models with Adam using L_2 regularization on all parameters. To have a good estimation of $P(\text{R})$ in the computation of $\mathcal{L}_{\text{D}}$, we use a batch size of 100. Our word embeddings are of size 50, entities embeddings of size $m = 10$. We sample $k = 5$ negative samples to estimate $\mathcal{L}_{\text{EP}}$. Lastly, we set $\alpha = 0.01$ and $\beta = 0.02$. All three datasets come with a validation set, and following Marcheggiani and Titov (2016), we used it for cross-validation to optimize the B^3 F_1.

3.3.3 Results

The results reported in Table 3.1 are the average test scores of three runs on the NYT + FB and T-REx SPO datasets, using different random initialization of the parameters—in practice, the variance was low enough so that reported results can be analyzed. We observe that regardless of the model and metrics, the highest measures are obtained on T-REx SPO, then NYT + FB and finally T-REx DS. This was to be expected since T-REx SPO was built to be easy, while hard-to-process sentences were filtered out of NYT + FB (Marcheggiani and Titov 2016; Yao et al. 2011). We also observe that the main metrics agree in general (B^3, V-measure and ARI) in most cases. Performing a PCA on the measures, we observed that V-measure forms a nearly-orthogonal axis to B^3, and to a lesser extent ARI. Hence we can focus on B^3 and V-measure in our analysis.

We first measure the benefit of our RelDist losses: on all datasets and metrics, the two models using $\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$ are systematically better than the ones using $\mathcal{L}_{\text{VAE REG}}$:

- The PCNN models consistently gain between 7 and 11 points in B^3 F_1 from these additional losses;
- The feature-based linear classifier benefits from the RelDist losses to a lesser extent, except on the T-REx DS dataset on which the Linear+ $\mathcal{L}_{\text{VAE REG}}$ model without the RelDist losses completely collapses—we hypothesize that this dataset is too hard for the model given the number of parameters to estimate.

Dataset	Model		B^3			V-measure			ARI
	Classifier	Reg.	F_1	Prec.	Rec.	F_1	Hom.	Comp.	
NYT + FB	rel-LDA		29.1	24.8	35.2	30.0	26.1	35.1	13.3
	rel-LDA1		36.9	30.4	47.0	37.4	31.9	45.1	24.2
	Linear	$\mathcal{L}_{\text{VAE REG}}$	35.2	23.8	67.1	27.0	18.6	49.6	18.7
	PCNN	$\mathcal{L}_{\text{VAE REG}}$	27.6	24.3	31.9	24.7	21.2	29.6	15.7
	Linear	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	37.5	31.1	47.4	**38.7**	32.6	47.8	27.6
	PCNN	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	**39.4**	32.2	50.7	38.3	32.2	47.2	**33.8**
	BERTcoder†	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	41.5	34.6	51.8	39.9	33.9	48.5	35.1
	BERTcoder†	SelfORE†	*49.1*	47.3	51.1	*46.6*	45.7	47.6	*40.3*
T-REx SPO	rel-LDA		11.9	10.2	14.1	5.9	4.9	7.4	3.9
	rel-LDA1		18.5	14.3	26.1	19.4	16.1	24.5	8.6
	Linear	$\mathcal{L}_{\text{VAE REG}}$	24.8	20.6	31.3	23.6	19.1	30.6	12.6
	PCNN	$\mathcal{L}_{\text{VAE REG}}$	25.3	19.2	37.0	23.1	18.1	31.9	10.8
	Linear	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	29.5	22.7	42.0	34.8	28.4	45.1	20.3
	PCNN	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	**36.3**	28.4	50.3	***41.4***	33.7	53.6	**21.3**
	BERTcoder†	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	38.1	30.7	50.3	39.1	37.6	40.8	23.5
	BERTcoder†	SelfORE†	*41.0*	39.4	42.8	*41.4*	40.3	42.5	*33.7*
T-REx DS	rel-LDA		9.7	6.8	17.0	8.3	6.6	11.4	2.2
	rel-LDA1		12.7	8.3	26.6	17.0	13.3	23.5	3.4
	Linear	$\mathcal{L}_{\text{VAE REG}}$	9.0	6.4	15.5	5.7	4.5	7.9	1.9
	PCNN	$\mathcal{L}_{\text{VAE REG}}$	12.2	8.6	21.1	12.9	10.1	18.0	2.9
	Linear	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	19.5	13.3	36.7	**30.6**	24.1	42.1	**11.5**
	PCNN	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	**19.7**	14.0	33.4	26.6	20.8	36.8	9.4
	BERTcoder†	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	22.4	17.6	30.8	31.2	26.3	38.3	12.3
	BERTcoder†	SelfORE†	*32.9*	29.7	36.8	*32.4*	30.1	35.1	*20.1*

Table 3.1: Results (percentage) on our three datasets. The results for rel-LDA, rel-LDA1, Linear and PCNN are our own, while results for BERTcoder and SelfORE, marked with †, are from X. Hu et al. (2020). The best results at the time of publication of our article are in **bold**, while the best results at the time of writing are in *italic*.

We now restrict to discriminative models based on $\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$. We note that both relation classifier (Linear and PCNN) exhibit better performances than generative ones (rel-LDA, rel-LDA1) with a difference ranging from 2.5/0.6 (NYT + FB, for Linear/PCNN) to 11/17.8 (on T-REx SPO). However, the advantage of PCNNs over feature-based classifiers is not completely clear. While the PCNN version has a systematically better B^3 F_1 on all datasets (differences of 1.9/6.8/0.2 respectively for NYT+FB/T-REx SPO/T-REx DS), the V-measure decreases by 0.4/4.0 on respectively NYT + FB/T-REx DS, and ARI by 2.1 on T-REx DS. As B^3 F_1 was used for validation, this shows that the PCNN models overfit this metric by polluting relatively clean clusters with unrelated sentences or degrades well clustered gold relations by splitting them into two clusters.

The BERTcoder classifier improves all metrics consistently, with the sole exception of the V-measure on the T-REx SPO dataset. This can be explained both by the larger expressive power of BERT and by its pretraining as a language model. The SelfORE model, which is built on top of a BERTcoder further improves the results on all datasets. Since these results are from a subsequent work (X. Hu et al. 2020), we won't delve too much into details. As mentioned in Section 2.5.7, SelfORE is an iterative algorithm; the $\mathscr{H}_{\text{UNIFORM}}$ assumption is enforced on the whole dataset at once, thus solving $\mathscr{P}2$. While to solve $\mathscr{P}1$, SelfORE uses a concentration objective (through the square in the target distribution $\boldsymbol{P}$ in Equation 2.11). While the BERTcoder can replace our PCNN classifier and can be evaluated

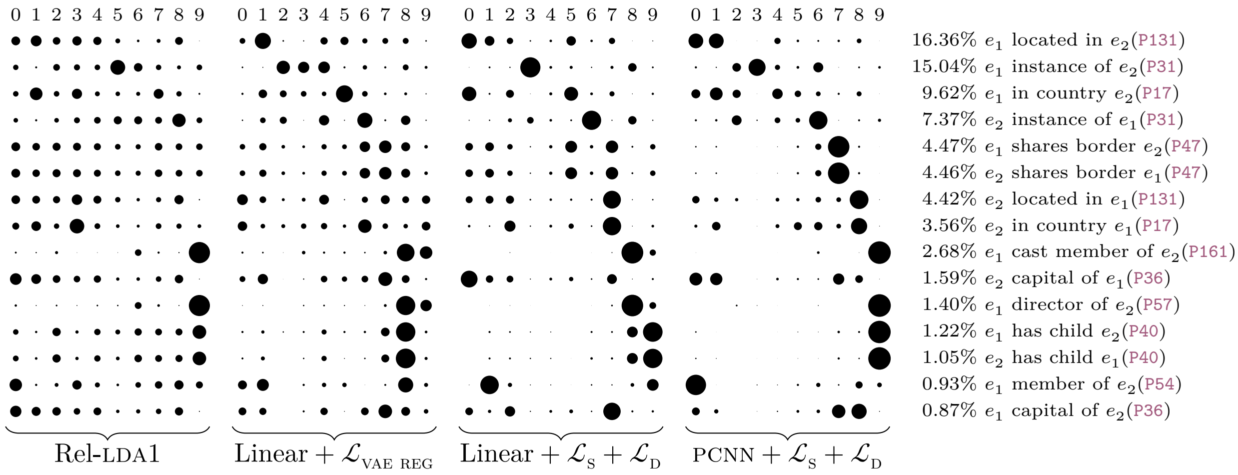


Figure 3.4: Normalized confusion matrices for the T-REx SPO dataset. For each model, each of the 10 columns corresponds to a predicted relation cluster, which were sorted to ease comparison. The rows identify Wikidata relations sorted by their frequency in the T-REx SPO corpus (reported as percentage in front of each relation name). The area of each circle is proportional to the number of sentences in the cell. For clarity, the matrix was normalized so that each row sum to 1, thus it is more akin to a B^3 per-item recall than a true confusion matrix.

with our regularization losses, the SelfORE algorithm is a replacement for the $\mathcal{L}_{\text{EP}} + \mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$ and can't be use jointly with $\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$. In theory, the SelfORE algorithm could be used with a linear or PCNN encoder. However, SelfORE strongly relies on a good initial representation; such a model would need to be pre-trained as a language model beforehand.

3.3.4 Qualitative Analysis

Since, for our model of interest, all the metrics agree on the T-REx SPO dataset, we plot the confusion matrix of our models in Figure 3.4. Each row is labeled with the gold Wikidata relation extracted through distant supervision. For example, the top left cell of each matrix correspond to the value $P(c(\mathrm{X}) = 0 \mid g(\mathrm{X}) = \text{“}e_1 \textit{ located in } e_2\text{”})$ using the notation of Section 2.5.1. Since relations are generally not symmetric, each Wikidata relation appears twice in the table, once for each disposition of the entities in the sentence. This is particularly problematic with symmetric relations such as “shares border,” which are two different gold relations that actually convey the same semantic relation.

To interpret Figure 3.4, we have to see whether a predicted cluster (column) contains different gold relations—paying attention to the fact that the most important gold relations are listed in the top rows (the top 5 relations account for 50% of sentences). The first thing to notice is that the confusion matrix of both models using our RelDist losses ($\mathcal{L}_{\text{S}}+\mathcal{L}_{\text{D}}$) are sparser (for each column), which means that our models better separate relations from each other. We observe that Linear + $\mathcal{L}_{\text{VAE REG}}$ (the model of the model of Marcheggiani and Titov 2016) is affected by the pitfall $\mathscr{P}1$ (uniform distribution) for many gold clusters. The $\mathcal{L}_{\text{VAE REG}}$ loss forces the classifier to be uncertain about which relation is expressed, translating into a dense confusion matrix and resulting in poor performances. The rel-LDA1 model is even worse and fails to identify clear clusters, showing the limitations of a purely generative approach that might focus on features not linked with any relation.

Focusing on our proposed model, PCNN$+\mathcal{L}_{\text{S}}+\mathcal{L}_{\text{D}}$ (rightmost figure), we looked at two different mistakes. The first is a gold cluster divided in two (low recall). When looking at clusters 0 and 1, we did not find any recognizable pattern. Moreover, the corresponding entity predictor parameters are very similar. This seems to be a limitation of the distance loss: splitting a large cluster in two may improve $\mathcal{L}_{\text{D}}$ but worsen all the evaluation metrics.

The model is then penalized by the fact that it lost one slot to transmit information between the classifier and the entity predictor. The second type of mistake is when a predicted cluster corresponds to two gold ones (low precision). Here, most of the mistakes seem understandable: “shares border” is symmetric (cluster 7), “located in” and “in country” (cluster 8) or “cast member” and “director of” (cluster 9) are clearly related. Note that other variants are also affected similarly, showing that the problem of granularity is complex.

3.4 Alternative Models

In this section, we present some variations we considered during the development of our model. However, we did not manage to obtain satisfactory results with these variants. When possible, we provide an analysis of why we think these variants did not work; keeping in mind that negative results are difficult to certify, poor results might be improved with a better hyperparameter search.

LSTM Relation Classifier Instead of a PCNN, we tried using a deep LSTM (Section 1.3.2.1) for our relation classifier. We never managed to obtain any results with them; the training always collapsed into one of $\mathscr{P}1$ or $\mathscr{P}2$. An LSTM is quite a lot harder to train than a CNN. The representation provided by an LSTM is the result of several non-linear operator compositions, through which it is hard to backpropagate information. On the other hand, with good initialization, the representation extracted by a CNN can be close to its input embeddings (which are pre-trained). Since the training of the entity predictor heavily depends on the relation classifier, it is not surprising that the training fails with an LSTM. The failure of the LSTM to provide a good representation at the beginning of the training procedure pushes the entity predictor to ignore the relation variable r, which therefore does not receive any gradient and thus does not provide any supervision back to the LSTM. Retrospectively, pre-training the sentence representation extractor with a language modeling loss could have overcome this problem. The initial representation would have been good enough for the entity predictor to provide some gradient back to the relation classifier. This is confirmed by the work of X. Hu et al. (2020), who trained a BERT relation classifier with our losses. In the end, what made a PCNN work is its shallowness and the pre-trained GloVe word embeddings.

Gumbel–Softmax Another approach to tackling $\mathscr{P}$ 1 (uniform output) would be to use a discrete distribution for the relation r; instead of marginalizing over all possible relations in Equation 3.3, we would only take the most likely relation. However, taking the maximum would not be differentiable. The Gumbel–softmax technique provides a solution to this problem. Let’s call $y_r \in \mathbb{R}$ for $r \in \mathcal{R}$ the unnormalized score assigned to each relation by the PCNN. It can be shown (Gumbel 1954) that sampling from $\operatorname{softmax}(\boldsymbol{y})$ is equivalent to taking $\operatorname{argmax}_{r\in\mathcal{R}} y_r + \mathrm{G}_r$ where G_r are randomly sampled from the Gumbel distribution. Knowing this, Jang et al. (2016) propose to use the following Gumbel–Softmax distribution:

Jang et al., “Categorical reparameterization with gumbel–softmax” ICLR 2016

$$\pi_r = \frac{(\exp(y_r) + \mathrm{G}_r) \;/\; \tau}{\sum_{r'\in\mathcal{R}} (\exp(y_{r'}) + \mathrm{G}_{r'}) \;/\; \tau}$$

$\mathscr{P}1$ solution	B^3			V-measure			ARI
	F_1	Prec.	Rec.	F_1	Hom.	Comp.	
$\mathcal{L}_\text{S}$ regularization	39.4	32.2	50.7	38.3	32.2	47.2	33.8
Gumbel–Softmax	35.0	29.9	42.2	33.2	28.3	40.2	25.1

Table 3.2: Quantitative results of the Gumbel–Softmax model on the NYT + FB dataset. The $\mathcal{L}_\text{S}$ solution is used together with $\mathcal{L}_\text{D}$ and a softmax activation, while the Gumbel–Softmax activation is used with $\mathcal{L}_\text{D}$ only. Therefore, the first row reports the same results present in Table 3.1.

This distribution has the advantage of being differentiable, barring the Gumbel variables G_r. Furthermore, when the temperature $\tau > 0$ is close to 1, this distribution looks like a standard softmax output. On the other hand, when the temperature is close to 0, this distribution is closer to a one-hot vector with low entropy. Decreasing the temperature gradually throughout the training process, this should help us solve $\mathscr{P}1$.

Following a grid search, we initially set $\tau = 1$ with an annealing rate of 0.9 per epoch. Table 3.2 compares the best Gumbel–Softmax results of $\mathcal{L}_\text{EP}+\mathcal{L}_\text{D}$ with the standard softmax result of $\mathcal{L}_\text{EP}+\mathcal{L}_\text{S}+\mathcal{L}_\text{D}$ discussed above. We do not use $\mathcal{L}_\text{S}$ with Gumbel–Softmax since both mechanisms seek to address $\mathscr{P}1$. While the Gumbel–Softmax prevents the model from falling entirely into $\mathscr{P}1$, it still underperforms compared to the $\mathcal{L}_\text{S}$ regularization of our standard model.

Aligning Sentences and Entity Pairs Another model we attempted to train purposes to align sentences and entities. It recombines our PCNN relation classifier with the energy function ψ into a new layout following a relaxation of the $\mathscr{H}_\text{PULLBACK}$ assumption.[53] In this model, we obtain a distribution over the relations $P(\text{r}_s \mid \text{blanked}(s))$ using a PCNN as described Section 3.1.1, but we also extract a distribution $P(\text{r}_e \mid \boldsymbol{e})$ using the energy function ψ normalized over the relations $P(r_e \mid e_1, e_2) \propto \exp(\psi(e_1, r_e, e_2))$. This model clearly assumes $\mathscr{H}_\text{PULLBACK}$ since it extracts a relation from the entities and from the sentence separately. However, in contrast to other models assuming $\mathscr{H}_\text{PULLBACK}$ (such as DIPRE, Section 2.3.2), we combine the separate relations into a single one to express the fact that a relation is both conveyed by the sentence and the entities:

[53] This hypothesis introduced Section 2.2.1 assumes that the relation can be found from the entities alone, and from the relations alone.

$$P(\text{r} = r \mid s, \boldsymbol{e}; \boldsymbol{\theta}, \boldsymbol{\phi}) = P(\text{r}_s = r \mid s; \boldsymbol{\phi}) P(\text{r}_e = r \mid \boldsymbol{e}; \boldsymbol{\theta}) \tag{3.9}$$

For the final prediction r, the assumption $\mathscr{H}_\text{PULLBACK}$ is not made, since it depends both on the sentence and entities. However, Equation 3.9 clearly assumes that r_s and r_e are independent and r does not capture any interaction between s and $\boldsymbol{e}$. To train this model, we force the two distributions to align by maximizing:

$$\mathcal{L}_\text{ALIGN}(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\log \sum_{r\in\mathcal{R}} P(r \mid s, \boldsymbol{e}; \boldsymbol{\theta}, \boldsymbol{\phi}) + \mathcal{L}_\text{D}(\boldsymbol{\theta}) + \mathcal{L}_\text{D}(\boldsymbol{\phi}). \tag{3.10}$$

For numerical stability, the first term of Equation 3.10 needs to be computed as:

$$\begin{aligned} -\log \sum_{r\in\mathcal{R}} P(r \mid s, \boldsymbol{e}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \\ -\log \sum_{r\in\mathcal{R}} \exp(y_r^{(s)} + y_e^{(s)}) \\ + \log \sum_{r\in\mathcal{R}} \exp(y_r^{(s)}) \\ + \log \sum_{r\in\mathcal{R}} \exp(y_r^{(e)}) \end{aligned}$$

where $\boldsymbol{y}^{(s)}$ and $\boldsymbol{y}^{(e)}$ are the logits used for predicting r_s and r_e respectively.

Here $\mathcal{L}_\text{S}$ is not needed since, in order to maximize the pointwise product of two probability mass functions, each distribution must be deterministic on a matching relation, which solves $\mathscr{P}1$.

We also attempted (without success) to align the two distribution by minimizing $\text{D}_\text{JSD}(\text{r}_s \parallel \text{r}_e)$. Where D_JSD is the Jensen–Shannon divergence defined as:

$$\begin{aligned} \text{D}_\text{JSD}(\text{r}_s \parallel \text{r}_e) = \frac{1}{2}\big(\text{D}_\text{KL}(\text{r}_s \parallel \text{m}) \\ + \text{D}_\text{KL}(\text{r}_e \parallel \text{m})\big) \end{aligned}$$

with $P(\text{m}) = \frac{1}{2}(P(\text{r}_s) + P(\text{r}_e))$.

Table 3.3 gives the results on the NYT + FB datasets and compares them to the fill-in-the-blank model of Section 3.1. The main problem we have with this model is its lack of stability. The average, maximum and minimum given in Table 3.3 are computed over eight runs. Similar results were observed with slightly different setups such as enforcing $\mathcal{L}_\text{D}$ on the product (r) instead of each distribution separately (r_s and r_e). As we can see, the alignment model sometimes reaches excellent performances

Model	B^3			V-measure			ARI
	F_1	Prec.	Rec.	F_1	Hom.	Comp.	
$\mathcal{L}_{\text{EP}} + \mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	39.4	32.2	50.7	38.3	32.2	47.2	33.8
$\mathcal{L}_{\text{ALIGN}}$ average	37.6	30.3	49.7	39.4	33.1	48.8	20.3
$\mathcal{L}_{\text{ALIGN}}$ maximum	41.2	33.6	53.4	43.5	36.9	53.1	29.5
$\mathcal{L}_{\text{ALIGN}}$ minimum	34.5	26.5	49.3	35.9	29.6	45.7	15.3

Table 3.3: Quantitative results of the alignment model on the NYT + FB dataset. The first row reports the same results present in Table 3.1. Eight alignment models were trained, the average scores are given in the second row, while the third and fourth rows report the best and worst model among the eight.

relative to the fill-in-the-blank model. However, this happens rarely, and on average, it performs more poorly according to the B^3 and ARI metrics. Its good V-measures scores are nevertheless encouraging.

3.5 Conclusion

In this chapter, we show that discriminative relation extraction models can be trained efficiently on unlabeled datasets. Unsupervised relation extraction models tend to produce impure clusters by enforcing a uniformity constrain at the level of a single sample. We proposed two losses (named RelDist) to effectively train expressive relation extraction models by enforcing the distribution over relations to be uniform—note that other target distributions could be used. In particular, we were able to successfully train a deep neural network classifier that only performed well in a supervised setting so far. We demonstrated the effectiveness of our RelDist losses on three datasets and showcased its effect on cluster purity.

While forcing a uniform distribution with the distance loss $\mathcal{L}_{\text{D}}$ might be meaningful with a low number of predicted clusters, it might not generalize to larger numbers of relations. Preliminary experiments seem to indicate that this can be addressed by replacing the uniform distribution in Equation 3.6 with the empirical distribution of the relations in the validation set or any other appropriate law if no validation set is available.[54] This would allow us to avoid the $\mathscr{H}_{\text{UNIFORM}}$ assumption.

All models presented in this chapter make extensive independence assumptions. As inferred in Section 3.4 and shown in subsequent work (X. Hu et al. 2020; Soares et al. 2019), this could be solved with sentence representations pre-trained with a language modeling task. Furthermore, the fill-in-the-blank model is inherently sentence-level. In the next chapter, we study how to build an unsupervised aggregate relation extraction model using a pre-trained BERTcoder.

[54] In practice, Zipf's law (described in the margin of Section 2.5.2) seems to fit the observed empirical distribution quite well.

Chapter 4

Graph-Based Aggregate Modeling

As we showcase in the last chapter, the relational semantics we are trying to model is challenging to capture in an unsupervised fashion. The information available in each sentence is scarce. To alleviate this problem, we can take a holistic approach by explicitly modeling the relational information at the dataset level, similarly to the aggregate approaches discussed in Section 2.4. The information encoded in the structure of the dataset can be modeled using a graph (Qian et al. 2019). In this chapter, we propose a graph-based unsupervised aggregate relation extraction method to exploit the signal in the dataset structure explicitly.

C'est même des hypothèses simples qu'il faut le plus se défier, parce que ce sont celles qui ont le plus de chances de passer inaperçues.

It is the simple hypotheses of which one must be most wary; because these are the ones that have the most chances of passing unnoticed.

— Henri Poincaré, *Thermodynamique (1908)*

In an extreme view, the world can be seen as only connections, nothing else. We think of a dictionary as the repository of meaning, but it defines words only in terms of other words. I liked the idea that a piece of information is really defined only by what it's related to, and how it's related. There really is little else to meaning. The structure is everything.

— Tim Berners-Lee, *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor* (1999)

Qian et al., "GraphIE: A Graph-Based Framework for Information Extraction" 2019

Kipf and Welling, "Semi-Supervised Classification with Graph Convolutional Networks" ICLR 2017

Since we model dataset-level information, we need to place ourselves in the aggregate setup (Section 2.1) as defined by Equation 2.2. As a reminder, the aggregate setup is in opposition to the sentential setup used in the previous chapter. In the sentential setup, we process sentences independently. In contrast, in the aggregate setup, we consider all the samples $\mathcal{D} \subseteq \mathcal{S} \times \mathcal{E}$ jointly to extract knowledge base facts $\mathcal{D}_{\text{KB}} \subseteq \mathcal{E} \times \mathcal{R}$, without necessarily mapping each individual sample to a fact. We already introduced two aggregate supervised relation extraction approaches relying on graph modeling, label propagation (Section 2.4.1) and EPGNN (Section 2.4.5). The latter uses a spectral graph convolutional network (GCN). GCNs are the main contribution coming from a recent resurgence of interest in graph-based approaches through the use of deep learning methods. It has been shown that these methods share some similarities with the Weisfeiler–Leman isomorphism test (Kipf and Welling 2017). A graph isomorphism test attempts to decide whether two graphs are identical. To this end, it assigns a color to each element, classifying it according to its neighborhood. Coupled with the assumption that sentences conveying similar relations have similar neighborhoods, this closely relates the isomorphism problem to unsupervised relation extraction. However, unsupervised GCNs are usually trained by assuming that neighboring samples have similar representations, completely discarding the characteristic of the Weisfeiler–Leman algorithm that makes it interesting from a relation extraction point of view. In this chapter, we propose alternative training objectives of unsupervised graph neural networks for relation extraction.

In Section 4.1, we see how to extend the definition of a simple graph to model a relation extraction problem. We then provide some statistics on the T-REx dataset in Section 4.2. The results support that large amount of information can be leveraged from topological features for the relation extraction problem. In Section 4.3, we take a quick tour of graph neural networks (GNN) and the Weisfeiler–Leman isomorphism test. Most GNNs apply to simple undirected graphs, whereas we need a more complex structure to encode the relation extraction task. While most models, such as EPGNN, try to adapt the encoding of relation extraction to simple undirected graphs, in Section 4.4, we propose to adapt existing GNN methods to the richer structure needed to fully capture the relation extraction problem. Finally, Section 4.5 presents the experimental results of the proposed approaches.

Notations used in this chapter. A simple undirected graph is defined as a tuple $G = (V, E)$ where V is a set of n vertices and E is a set of m edges.[55] An edge $\{u, v\} \in E$ connects two vertices $u, v \in V$, which are then said to be *neighbors*. We use $N : V \to 2^V$ to denote the function which associates to each vertex the set of its neighbors $N(u) = \{v \in V \mid \exists\{u, v\} \in E\}$. Alternatively, a graph G can be represented by its adjacency matrix $\boldsymbol{M} \in \{0, 1\}^{n\times n}$, with $m_{uv} = 1$ if $\{u, v\} \in E$ and $m_{uv} = 0$ otherwise. A graph is said to encode an adjacency relation on its vertices, which foreshadows the remainder of this chapter.

[55] In a simple graph, we always have $m \leq n(n-1)$ which tightens to $m \leq n(n-1)\,/\,2$ for undirected ones.

4.1 Encoding Relation Extraction as a Graph Problem

In this section, we describe how to frame the relation extraction problem as a problem on graphs. In particular, we describe the structure of an attributed multigraph which is a generalization of the simple undirected graph defined in the previous paragraph. This structure is needed to model entities linked by multiple relations or sentences since this can't readily be done with a simple graph.

The distinction between E and $\mathcal{E}$ is important. We decided to keep the usual $G = (V, E)$ notation for undirected graphs. However, the multigraph we describe in this section has the set of entities $\mathcal{E}$ as vertices. This set $\mathcal{E}$ takes the place of V; despite the similar notation, it has nothing to do with E.

Since a knowledge base relation can be formally defined as a set of entity pairs (Section 1.4.1), we can represent it using a single graph $G = (V, E)$ where V is the set of entities ($V = \mathcal{E}$) and E is the set of pairs linked by the relation ($E \in \mathcal{R}$). However, to encode the relation extraction task on a graph, we need different kinds of edges. We, therefore, use the structure of an attributed[56] multigraph $G = (\mathcal{E}, \mathcal{A}, \boldsymbol{\varepsilon}, \rho, \varsigma)$ where:[57]

- $\mathcal{E}$ is the set of entities, which corresponds to the vertices of G (indeed $\mathcal{E} = V$),
- $\mathcal{A}$ is the set of arcs, which represent a directed[58] link (usually a sentence) between two entities (this approximately corresponds to the set of edges E in a simple graph, but can also be seen as equivalent to a supervised set of samples $\mathcal{D}_\mathcal{R}$),
- $\varepsilon_1 : \mathcal{A} \to \mathcal{E}$ assigns to each arc its source vertex (the entity e_1),
- $\varepsilon_2 : \mathcal{A} \to \mathcal{E}$ assigns to each arc its target vertex (the entity e_2),
- $\varsigma : \mathcal{A} \to \mathcal{S}$ assigns to each arc $a \in \mathcal{A}$ the corresponding sentence containing $\varepsilon_1(a)$ and $\varepsilon_2(a)$,
- $\rho : \mathcal{A} \to \mathcal{R}$ assigns to each arc $a \in \mathcal{A}$ the relation linking the two entities conveyed by $\varsigma(a)$.

[56] The term "*labeled*" is usually reserved for graphs where the domain of attributes is discrete and finite. However the set of possible sentences $\mathcal{S}$ is not (theoretically) finite.

[57] To be perfectly formal, G should also depend on $\mathcal{S}$ and $\mathcal{R}$, the codomains of ς and ρ. We omit them for conciseness.

[58] We use the word *edge* to refer to a symmetric connection $\{u, v\}$, while *arc* refers to an asymmetric connection (u, v). Using this nomenclature, an undirected graph has *edges* while a directed graph has *arcs*.

In this graph, the vertices are entities with an arc linking them for each sentence in which they both appear. Figure 4.1 shows the graph corresponding to the sentences in Table 2.1. Let's call $a \in \mathcal{A}$ the highlighted bottom left arc in Figure 4.1 linking SMERSH to counterintelligence. Applying the above definitions to this arc we have:

- $\varepsilon_1(a) = \text{SMERSH}$ (Q158363)
- $\varepsilon_2(a) = \text{counterintelligence}$ (Q501700)
- $\varsigma(a) =$ In its $\underline{\text{counter-espionage}}_{e_2}$ and counter-intelligence roles, $\underline{\text{SMERSH}}_{e_1}$ appears to have been extremely successful throughout World War II.
- $\rho(a) =$ *field of work* (P101)

Remember that $\mathcal{S}$ is not simply a set of regular sentences but a set of sentences with two tagged and ordered entities.

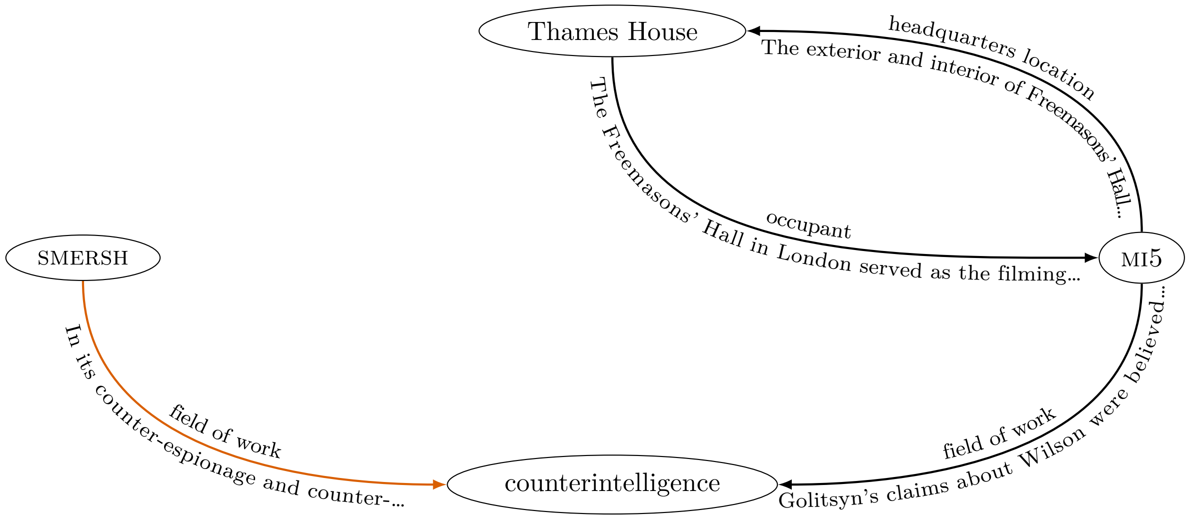


Figure 4.1: Multigraph G corresponding to the four samples of Table 2.1. For each arc a, its relation $\rho(a)$ is written over the arc, and the beginning of the conveying sentence $\varsigma(a)$ is written under the arc. For ease of reading, surface forms are given instead of numerical identifiers. The highlighted arc corresponds to the example given above.

In the supervised relation extraction task, the set of relations $\mathcal{R}$ is fully known, and ρ is partially known; the goal is to complete ρ. In the unsupervised relation extraction task, $\mathcal{R}$ is unknown, and ρ must be built from the ground up. We can also encode a knowledge base using this structure by removing the associated sentences (i.e. the ς attributes).[59]

Note that the graph G is directed because most relations and sentences are asymmetric (inverting the two entities changes the meaning). This is the only semantic associated with orientation.[60] In the unsupervised setting, when the graph is not labeled with relations, each arc $u \xrightarrow{s} v$ has a symmetric arc $u \xleftarrow{\breve{s}} v$ where $\breve{s} \in \mathcal{S}$ is the same sentence as $s \in \mathcal{S}$ with the tags $\_\__{e_1}$ and $\_\__{e_2}$ inverted.

For ease of notation, let us define the incident function $\mathcal{I}$ associating to each vertex its set of incident arcs $\mathcal{I}(e) = \{a \in \mathcal{A} \mid \varepsilon_1(a) = e \vee \varepsilon_2(a) = e\}$. In other words, $\mathcal{I}$ associates to each entity the set of samples in which it appears. Furthermore, for each relation $r \in \mathcal{R}$, we define the relation graphs $G_{\langle r \rangle} = (\mathcal{E}, \mathcal{A}_{\langle r \rangle}, \varepsilon_1, \varepsilon_2, \rho, \varsigma)$ where $\mathcal{A}_{\langle r \rangle} = \{a \in \mathcal{A} \mid \rho(a) = r\}$ is the set of arcs labeled with relation r. We can then define the out-neighbors $N_{\overrightarrow{\langle r \rangle}}$ and in-neighbors $N_{\overleftarrow{\langle r \rangle}}$ functions on the relation graph $G_{\langle r \rangle}$ as follows:[61]

$$N_{\overrightarrow{\langle r \rangle}}(e_1) = \{\, e_2 \in \mathcal{E} \mid \exists a \in \mathcal{A} : \varepsilon_1(a) = e_1 \wedge \varepsilon_2(a) = e_2 \wedge \rho(a) = r \,\},$$
$$N_{\overleftarrow{\langle r \rangle}}(e_1) = \{\, e_2 \in \mathcal{E} \mid \exists a \in \mathcal{A} : \varepsilon_2(a) = e_1 \wedge \varepsilon_1(a) = e_2 \wedge \rho(a) = r \,\}.$$

Using these definitions we can write expressions for the generic neighbors function:

$$N_{\langle r \rangle}(e) = N_{\overrightarrow{\langle r \rangle}}(e) \cup N_{\overleftarrow{\langle r \rangle}}(e),$$
$$N(e) = \bigcup_{r \in \mathcal{R}} N_{\langle r \rangle}(e).$$

[59] Indeed, in this case, the graph is simply a set of entities linked by relation arcs such as Sanaa $\xrightarrow{\textit{capital of}}$ Yemen.

[60] For example, while the notion of sink—a vertex with no outgoing arcs—might be of interest to graph theorists, it bears no special meaning in our encoding.

[61] Note that the functions we define here are for the open neighborhood. This means that we don't consider a vertex to be its own neighbor.

Finally, we can define the degree of a vertex as its number of neighbors:

$$\deg(e) = |N(e)|,$$

which can be broken down into in-degree and out-degree using in-neighbors and out-neighbors.

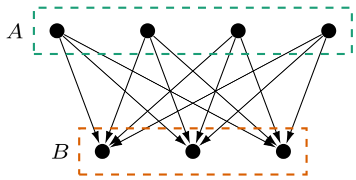
Using these notations we can reformulate modeling assumptions such as $\mathscr{H}_{\text{BICLIQUE}}$ (Section 2.5.4), $\mathscr{H}_{\text{1-ADJACENCY}}$ (Section 2.3.2) and $\mathscr{H}_{1\to 1}$ (Section 2.5.6). For example, the hypothesis $\mathscr{H}_{\text{BICLIQUE}}$ draw its name from the fact that for all relation $r \in \mathcal{R}$, the relation graph $G_{\langle r\rangle}$ is assumed to be a biclique.[62] This is especially of interest to study matching the blanks (MTB, Section 2.5.6). It can be analyzed using the following graph:

$$\begin{tikzcd} e_3 & e_1 \arrow[l, "r_3"'] \arrow[r, bend left, "r_1"] \arrow[r, bend right, "r_2"'] & e_2 \end{tikzcd}$$

MTB makes two main assumptions: $\mathscr{H}_{\text{1-ADJACENCY}}$ and $\mathscr{H}_{1\to 1}$. In the above graph, $\mathscr{H}_{\text{1-ADJACENCY}}$ implies that r_1 and r_2 should be the same, while $\mathscr{H}_{1\to 1}$ implies that r_3 should be different from r_1 and r_2. From this simple example, we can also see that MTB training is 1-localized, which means that it only exploits the fact that two samples are direct neighbors.[63] In contrast, a sentential approach is 0-localized; it completely ignores other samples. This is actually the case of MTB during evaluation. The same problem plagues the fill-in-the-blank model of Chapter 3; while training is influenced by the direct neighbors (through the entity embeddings), when classifying an unknown sample, its neighbors are ignored. The goal of this chapter is to consider larger neighborhoods both for training unsupervised models and for making predictions with them.

Since we mention several hypotheses, we take this opportunity to remind the reader that all assumptions are detailed in Appendix B.

62 A biclique is a *complete bipartite graph*. Its vertices can be split into two sets $A, B \subseteq \mathcal{E}$ such that each vertex in A is linked to all vertices in B. For example:



63 Here we use *neighbors* as in "arc-neighbors." This is a relation between two arcs sharing a common endpoint. Arc-neighbors are simple neighbors in the line graph described in Section 4.4.1.

4.2 Preliminary Analysis and Proof of Principle

In this section, we want to ensure the soundness of graph-based approaches by providing some statistics about a large relation extraction dataset. In particular, we start by building an attributed multigraph as described in Section 4.1. We focus on T-REx (Section C.7, Elsahar et al. 2018), an alignment (Section 2.2.2) of Wikipedia with Wikidata. This dataset has the advantage of being both large and publicly available. Note that the graph we analyze in this section is not a knowledge base. Each arc is both labeled with a relation and attributed with a sentence. The fact that several arcs are incident to a vertex does not necessarily imply that the corresponding entity is linked by several relations, only that it was mentioned multiple times.

Figure 4.2 shows the distribution of vertices' degrees in the graph associated with T-REx. The first thing we can notice about this graph is that it is *scale-free*. This means that a random vertex $v \in \mathcal{E}$ has degree $\deg(v) = k$ with probability $P(k) \propto k^{-\gamma}$ for a parameter γ which depends on the graph. In other words, the distribution of degrees follows a power law. In a scale-free graph, a lot of vertices have few neighbors. In contrast, the distribution of degrees in a random Erdős–Rényi graph[64] is expected to follow a binomial distribution. Scale-free graphs occur in a number of

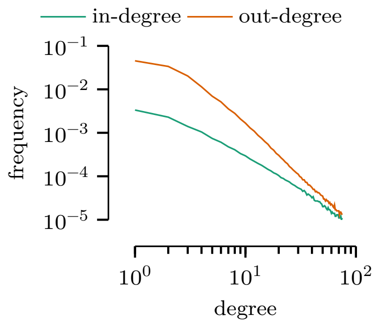


Figure 4.2: T-REx vertices degree distribution. The lines give the frequency of vertices with the given in- and out-degree in the dataset. Note that both axes are log-scaled. This plot was cut at a degree of 75, which corresponds to a minimum frequency of 10^{-5} out of a total of 19 392 185 arcs. In reality, the vertex with the maximum degree is "United States of America" Q30 with an in-degree of 1 522 224. The asymmetry between the distribution of in-degrees and out-degrees can be explained by the fact that knowledge bases prefer to encode many-to-one relations instead of their one-to-many converse.

64 There are several different ways to sample random graphs; the Erdős–Rényi model is one of them. In this model, arcs are incrementally added between two uniformly chosen vertices. In contrast, if vertices with already high degrees are selected more often (the Barabási–Albert model), the resulting graph is scale-free.

contexts, such as social networks and graphs of linked web pages. Most unsupervised relation extraction datasets and knowledge bases should be expected to be scale-free. This needs to be kept in mind when designing graph-processing algorithms for relation extraction. Indeed most vertices have a small neighborhood, so we might be tempted to take neighbors of neighbors carelessly. However, scale-free graphs have a very small diameter[65] $D \in O(\log \log n)$. This means that we can quickly reach most vertices following a small number of arcs. This is in part due to the fact that some vertices have very high degree, for example in T-REx, the vertex "United States of America" `Q30` is highly connected with $\deg(\texttt{Q30}) = 1\,697\,334$. In particular, this implies that by considering neighbors of neighbors, we quickly need to consider the whole graph; this is particularly problematic for graph convolutional networks described in Section 4.3.

[65] The diameter of a graph is the length of the longest shortest-path:

$$D = \max_{u,v \in \mathcal{E}} \delta(u, v),$$

where $\delta(u, v)$ is the length of the shortest path from u to v.

We now come to the main incentive for taking a graph-based approach to the unsupervised relation extraction task:

Hypothesis: *In the relation extraction problem, we can get additional information from the neighborhood of a sample.*

To test this hypothesis, we compute statistics on the distribution of neighbors. However, as we just saw, the support of this distribution is of high dimension. Hence, we look at the statistics of paths in our multigraph.[66] As a graph theory reminder, we can formally define a path as follows:

- A *walk* on length n is a sequence of arcs $a_1, a_2, \dots, a_n \in \mathcal{A}$ such that $\varepsilon_2(a_{i-1}) = \varepsilon_1(a_i)$ for all $i = 2, \dots, n$.
- A *trail* is a walk with $a_i \neq a_j$ for all $1 \leq i < j \leq n$ (arcs do not repeat). In practice this means that $(s, \boldsymbol{e})$ do not repeat. It is not a statement about relations conveyed by these arcs; it is entirely possible that for some i, j we have $\rho(a_i) = \rho(a_j)$.
- A *path* is a trail with $\varepsilon_1(a_i) \neq \varepsilon_1(a_j)$ for all $1 \leq i < j \leq n$ (vertices do not repeat).

[66] Paths of length k are in a domain of size $|\mathcal{R}|^k$, whereas neighbors are in a domain of size $|\mathcal{R}|^{\Delta(G)}$ with $\Delta(G)$ designating the maximum degree in G. By studying paths of length 3, we are effectively studying a subsampled neighborhood of the central arc.

It is also possible to base these definitions on *open walks*, which are walks where $\varepsilon_1(a_1) \neq \varepsilon_2(a_n)$ (the walk does not end where it started). We base the discussion of this section around the following random path:

$$\mathrm{e}_1 \xrightarrow{\mathrm{r}_1} \mathrm{e}_2 \xrightarrow{\mathrm{r}_2} \mathrm{e}_3 \xrightarrow{\mathrm{r}_3} \mathrm{e}_4,$$

Using these definitions, we can restate our hypothesis. In this path, we expect $\mathrm{r}_2 \not\perp\!\!\!\perp \mathrm{r}_1$ and $\mathrm{r}_2 \not\perp\!\!\!\perp \mathrm{r}_3$. However, enumerating all possible paths in a graph with $n = 2\,819\,966$ vertices and $m = 19\,392\,185$ arcs is not practical.

The symbol $\not\perp\!\!\!\perp$ is used to mean "not independent":

$$\mathrm{a} \not\perp\!\!\!\perp \mathrm{b} \iff P(\mathrm{a}, \mathrm{b}) \neq P(\mathrm{a})P(\mathrm{b})$$

To approximate path statistics, we turn to sampling. However, uniformly sampling paths is not straightforward. As a first intuition, to uniformly sample a path of length 1—that is, an arc—we can use the following procedure:

1. Sample an entity e_1 weighted by its degree,
   $e_1 \sim \operatorname{Cat}(\mathcal{E}, e \mapsto \deg(e) \mathbin{/} 2m)$
2. Uniformly sample an arc incident to the entity e_1.
   $a \sim \mathcal{U}(\mathcal{I}(e_1))$

$\operatorname{Cat}(\mathcal{E}, f)$ refers to the Categorical distribution over the set $\mathcal{E}$ where the probability of picking $e \in \mathcal{E}$ is $f(e)$. The $2m$ appears from the normalization factor $\sum_{e \in \mathcal{E}} \deg(e) = 2m$.

The first vertex we select must be weighted by how many paths start there, and since paths of length 1 are arcs, we weight each vertex by its degree.[67] If we want to sample paths of length 2, the first vertex must be selected according to the number of paths of length 2 starting there. Then the second vertex is selected among the neighbors of the first weighted by the number of paths of length 1 starting there, etc.

[67] To give an intuition, we can also think of what would happen if we chose both the entity and incident arc uniformly. An arc that links two entities otherwise unrelated to any other entities is likely to be sampled since sampling any of its two endpoints as e_1 would guarantee we select this arc. On

the other hand, an arc whose both endpoints have high degrees has little chance of being sampled since even if one of its endpoints is selected as e_1 in the first step, the arc is unlikely to be selected in the second step.

algorithm PATH COUNTING
  Inputs: $G = (\mathcal{E}, \mathcal{A}, \boldsymbol{\varepsilon}, \rho, \varsigma)$ relation multigraph
    k paths length
  Output: C relation paths counter

  ▷ *Initialization* ◁
  $C \leftarrow$ new counter $\mathcal{R}^k \to \mathbb{R}$ initialized at 0
  ▷ *Main Loop* ◁
  loop
    ▷ *Initialize the importance weight with* $\mathcal{W}^k$ ◁
    $w \leftarrow \left(\mathbf{1}^\intercal \boldsymbol{M}^k \mathbf{1}\right)^{-1}$ ▷ *$\boldsymbol{M}$ is the adjacency matrix*
    Initialize empty walk $\boldsymbol{a} = ()$
    Sample $v \sim \mathcal{U}(\mathcal{E})$
    $w \leftarrow n \times w$ ▷ *Update w following the sampling of v*
    for $i = 1, \ldots, k$ **do**
      Sample $x \sim \mathcal{U}(\mathcal{I}(v))$
      $w \leftarrow w \times \deg(v)$ ▷ *Accumulate $1 \,/\, \mathcal{F}^k$*
      if $\varepsilon_1(x) = v$ **then** ▷ *Continue with $\boldsymbol{\varepsilon}(x) \setminus \{v\}$*
        Append x to $\boldsymbol{a}$
        $v \leftarrow \varepsilon_2(x)$
      else
        Append $\breve{x}$ to $\boldsymbol{a}$
        $v \leftarrow \varepsilon_1(x)$
    if $\boldsymbol{a}$ is a path **then**
      $\boldsymbol{r} \leftarrow (\rho(a_i))_{1 \leq i \leq k}$ ▷ *Take the relations of $\boldsymbol{a}$*
      $C[\boldsymbol{r}] \leftarrow C[\boldsymbol{r}] + w$
  output C

Algorithm 4.1: Path counting algorithm. The higher the number of iterations of the main loop, the more precise the results will be. In our experiments, we used one billion iterations. The inner for loop builds the walk $\boldsymbol{a}$. If it is a correct path, the relation type of the path is added to the counter with importance weight w. For numerical stability, we actually compute w in log-space. The initial factor $n = |\mathcal{E}|$ in w comes from the preceding uniform sampling of v from $\mathcal{E}$, which is part of the computation of $\mathcal{F}^k$.

Sadly enough, counting paths is #P-complete[68] (Valiant 1979) so we must rely on the regularity of our graph and turn to approximate algorithms. We propose to use the number of walks as an approximation of the number of paths.[69] A classical result on simple graphs $G = (V, E)$ is that the powers of the adjacency matrix $\boldsymbol{M}$ count the number of walks between pairs of vertices. For any two vertices $u, v \in V$, the value m_{uv}^k—to be interpreted as $(\boldsymbol{M}^k)_{uv}$—is the number of walks of length k from u to v. In the case of our multigraph, if we wish to count walks, the adjacency matrix should contain the number of arcs—that is, the number of walks of length 1—between vertices.

We could then build a Monte Carlo estimate by following the naive procedure above of sampling vertices one by one according to the number of walks starting with them. Let's call $\mathcal{W}^k$ this distribution over walks of length k. Sampling from $\mathcal{W}^k$ is particularly slow since it involves sampling from a categorical distribution over thousands of elements. Since we only want to evaluate a (counting) function over an expectation $\mathbb{E}_{\boldsymbol{a} \sim \mathcal{W}^k}$, we can instead perform importance sampling. We use the substitute distribution $\mathcal{F}^k$ that uniformly selects a random neighbor at each step. To make this trick work, we only need to compute the importance weights $\frac{\mathcal{W}^k(\boldsymbol{a})}{\mathcal{F}^k(\boldsymbol{a})}$ for all walks $\boldsymbol{a} \in \mathcal{A}^k$. Since $\mathcal{W}^k$ is the uniform distribution over all walks, it is constant $\mathcal{W}^k(\boldsymbol{a}) = (\mathbf{1}^\intercal \boldsymbol{M}^k \mathbf{1})^{-1}$. On the other hand $\mathcal{F}^k(\boldsymbol{a})$ can be trivially computed as the product of inverse degrees of a_i. The resulting counting procedure is listed as Algorithm 4.1. We still need to reject non-paths at the end of the main loop. Note that this algorithm is not exact since the importance weights w are computed from the number of walks, not paths.

Using this algorithm on one billion samples from T-REx, we find that

[68] A functional complexity class at least as hard as NP-complete.

[69] Other approximations of path counting exist (Roberts and Kroese 2007), but the approach we propose is particularly suited to our multigraph. In particular, the shape parameter γ of our degree distribution is relatively small, which produces a large number of outliers. Our importance-sampling-based approach allows us to reduce the variance of the frequency estimations.

Frequency	Relation path	
	Surface forms	Identifiers
54.657‰	*country* • *diplomatic relation* • $\overparen{\textit{country}}$	P17 • P530 • $\overparen{\text{P17}}$
31.696‰	*country* • *diplomatic relation* • $\overparen{\textit{citizen of}}$	P17 • P530 • $\overparen{\text{P27}}$
6.680‰	*country* • *shares border with* • $\overparen{\textit{citizen of}}$	P17 • P47 • $\overparen{\text{P27}}$
0.013‰	*country* • *seceded from* • $\overparen{\textit{citizen of}}$	P17 • P807 • $\overparen{\text{P27}}$
9.445‰	*sport* • $\overparen{\textit{sport}}$ • $\overparen{\textit{member of}}_{\text{ST}}$	P641 • $\overparen{\text{P641}}$ • $\overparen{\text{P54}}$
10^{-6} ‰	*sport* • $\overparen{\textit{industry}}$ • $\overparen{\textit{member of}}_{\text{ST}}$	P641 • $\overparen{\text{P452}}$ • $\overparen{\text{P54}}$

Table 4.1: Frequencies of some paths of length 3 in T-REx. The first column gives the approximate per mille frequency of paths with the given type. It is computed as the importance weight attributed to the path by the counter C in Algorithm 4.1 divided by the sum of all importance weights in C. We use $_{\text{ST}}$ as an abbreviation of "sport team." The path in the first row is the most frequent one in the dataset; other paths were selected for illustrative purposes. The last path was sampled a single time with an importance weight of 0.89.

the most common paths of length three are related to geopolitical relations,[70] see Table 4.1. Let us now turn to statistics that could help relation extraction models. To showcase the dependency between a sample's relation r_2 and its neighbors r_1 and r_3, we investigate the distribution $P(\mathrm{r}_2 \mid \mathrm{r}_1, \mathrm{r}_3)$. In other words, given a sample, we want to see how its relation is influenced by the relations of two neighboring samples.

The first value we can look at is the entropy[71] $\mathrm{H}(\mathrm{r}_2 \mid r_1, r_3)$. For example, in the case of $r_1 = \textit{sport}$ and $r_3 = \overparen{\textit{member of}}_{\text{ST}}$, all observed values of r_2 are given in Table 4.1. All of them were $\overparen{\textit{sport}}$ with the exception of a single path, which means that $\mathrm{H}(\mathrm{r}_2 \mid r_1, r_3) \approx 0$. In other words, if we are given a sample $(s, \boldsymbol{e}) \in \mathcal{D}$ and we suspect another sentence containing e_1 to convey *sport* and another containing e_2 to convey $\overparen{\textit{member of}}_{\text{ST}}$, we can be almost certain that the sample $(s, \boldsymbol{e})$ conveys $\overparen{\textit{sport}}$.

To measure this type of dependency at the level of the dataset, we can look at the following value:

$$\mathrm{D}_{\text{KL}}\left(P(\mathrm{r}_2 \mid r_1, r_3) \parallel P(\mathrm{r}_2)\right)$$

The Kullback–Leibler divergence is also called the *relative entropy*. Indeed, $\mathrm{D}_{\text{KL}}(P \parallel Q)$ can be interpreted as the additional quantity of information needed to encode P using the (suboptimal) entropy encoding given by Q. If this value is 0, it means that no additional information was provided by r_1 and r_3. When marginalizing over all possible contexts r_1, r_3, we obtain the mutual information between the relation of a sample r_2 and the relation of two of its neighbors. On T-REx, we observe:

$$\mathrm{I}(\mathrm{r}_2; \mathrm{r}_1, \mathrm{r}_3) \approx 6.95 \text{ bits}$$

In other words, we can gain 6.95 bits of information simply by modeling two neighbors (one per entity). These 6.95 bits can be interpreted as the number of bits needed to perfectly encode r_2 given r_1, r_3 (the conditional entropy $\mathrm{H}(\mathrm{r}_2 \mid \mathrm{r}_1, \mathrm{r}_3) \approx 1.06$ bits) substracted from the number of bits needed to encode r_2 without looking at its neighbors (the cross-entropy $\mathbb{E}_{r_1, r_3}[\mathrm{H}_{P(\mathrm{r}_2)}(\mathrm{r}_2 \mid r_1, r_3)] \approx 8.01$ bits).[72] In other words, most of the uncertainty about the relation of a sample can be removed by looking at the relations of two of its neighbors.

[70] This is not surprising as most general knowledge datasets are dominated by geopolitical entities and relations.

[71] This is not a conditional entropy. The context relations r_1, r_3 are fixed; they correspond to elementary events, not random variables (as shown by the fact that they are italicized, not upshape).

As a reference for the remainder of this section, the distribution of relation in T-REx has an entropy of $\mathrm{H}(\mathrm{r}) \approx 6.26$ bits. This is for a domain of $|\mathcal{R}| = 1\,316$ relations.

To give a first intuition of what this value represents, we take once again the trivial example of r_1 = *sport* and r_3 = $\overparen{\textit{member of}}_{\text{ST}}$. In this case, $\mathrm{D}_{\text{KL}}\left(P(\mathrm{r}_2 \mid r_1, r_3) \parallel P(\mathrm{r}_2)\right) \approx 5.47$ bits. This is due to the fact that encoding r_2 given its neighbors necessitates close to 0 bits (as shown in Table 4.1, r_2 almost always takes the value $\overparen{\textit{sport}}$) but encoding $\overparen{\textit{sport}}$ among all possible relations in $\mathcal{R}$ necessitates 5.47 bits (which is a bit less than most relations since $\overparen{\textit{sport}}$ commonly appears in T-REx).

[72] We denote the cross-entropy by $\mathrm{H}_Q(P) = -\mathbb{E}_P[\log Q]$.

4.3 Related Work

In the previous section, we show that the attributed multigraph encoding we introduced in Section 4.1 can help us leverage additional information for the relation extraction task. In this section, we present the existing framework for computing distributed representations of graphs. In most cases, these process simple undirected graphs $G = (V, E)$. Still, these methods are applicable to our relation extraction multigraph with some modifications, as shown in Sections 4.3.4 and 4.4.

The use of graphs in deep learning has seen a recent surge of interest over the last few years. This produced a set of models known as graph neural networks (GNN) and graph convolutional networks (GCN).[73] While the first works on GNN started more than twenty years ago (Sperduti and Starita 1997), we won't go into a detailed historical review, and we exclusively focus on recent models. Note that we already presented an older graph-based approach in Section 2.4.1, the label propagation algorithm. We also discussed EPGNN in Section 2.4.5, which is a model built on top of a GCN. We further draw parallels between EPGNN and our proposed approach in Section 4.4.1.

[73] The term GCN is used with different meanings by various authors. GCNs are always GNNs, but the reverse is not true. However, in practice, the GNNs we describe in this section can essentially be described as GCNs. We use the term GCN to describe models whose purpose is to have a similar function on graphs as CNNs have on images. Some authors only refer to the model of Kipf and Welling (2017) described in Section 4.3.2 as a GCN. In this case, what we call GCN can be called convGNN (convolutional graph neural networks). In any case, GNN and GCN can be considered almost synonymous for the purpose of this thesis since we don't describe any exotic GNN which clearly falls outside of the realm of GCN.

The thread of reasoning behind this section is as follows:

- We present the "usual" way to process graphs (Sections 4.3.1–4.3.4).
- We present the theory behind these methods (Section 4.3.5).
- We show how this theoretical background can help us design a new approach specific to the unsupervised relation extraction task (Section 4.4).

In this related work overview, we mainly describe algorithms working on standard $G = (V, E)$ graphs, not the labeled multigraphs of Section 4.1, with the exception of Section 4.3.4. We start by quickly describing models based on random walks in Section 4.3.1; these are spatial methods which serve as a gentle introduction to the manipulation of graphs by neural networks. Furthermore, they were influential in the development of subsequent models and in our preliminary analysis with computation of path statistics (Section 4.2), which allows us to draw parallels with more modern approaches. We then introduce the two main classes of GCN—which consequently are also the two main classes of GNN—used nowadays: spectral (Section 4.3.2) and spatial (Section 4.3.3). Apart from the few works mentioned in Chapter 2, GNNs were seldom used for relation extraction. We, therefore, focus on the evaluation of GNN on an entity classification task, which while different from our problem, works on similar data. In Section 4.3.4, we describe models designed to handle relational data in a knowledge base, in particular R-GCN. We close this related work with a presentation of the Weisfeiler–Leman isomorphism test in Section 4.3.5; it serves as a theoretical motivation behind both GCNs and our proposed approach.

4.3.1 Random-Walk-Based Models

DeepWalk (Perozzi et al. 2014) is a method to learn vertex representations from the structure of the graph alone. The representations encode how likely it is for two vertices to be close to each other in the graph. To this end, DeepWalk models the likelihood of random walks in the graph (Section 4.2). These walks are simply sequences of vertices. To obtain a distributed representation out of them, we can use the NLP approaches of Sections 1.2 and 1.3 by treating the set of vertices as the vocabulary $V = \mathcal{E}$. In particular, DeepWalk uses the skip-gram model of Word2vec (Section 1.2.1.1), using hierarchical softmax to approximate the partition function over all words—i.e. vertices. Vertices part of the same random walk are used as positive examples. In the same way that learning rep-

Perozzi et al., "DeepWalk: Online Learning of Social Representations" KDD 2014 resentations to predict the neighborhood of a word gives good word representations, modeling the neighborhood of a vertex gives good vertex representations.

Perozzi et al. (2014) evaluate their model on a node classification task. For example, one of the datasets they use is BlogCatalog (Tang and Liu 2009), where vertices correspond to blogs, edges are built from social network connections between the various bloggers, and predicted labels are the set of topics on which each blog focuses. DeepWalk is a transductive method but was extended into an inductive approach called planetoid (Yang et al. 2016). Planetoid also proposes an evaluation on an entity classification task performed on the NELL dataset. The goal of this task is to find the type of an entity (e.g. person, organization, location…) in a knowledge base (Section 1.4). To this end, a special bipartite[74] graph $G_\text{B} = (V_\text{B}, E_\text{B})$ is constructed where $V_\text{B} = \mathcal{E} \cup \mathcal{R}$ and:

Yang et al., "Revisiting Semi-Supervised Learning with Graph Embeddings" ICML 2016

[74] A bipartite graph is a graph $G = (V, E)$ where the vertices can be split into two disjoint sets $V_1 \cup V_2 = V$ such that all edges $e \in E$ have one endpoint in V_1 and one endpoint in V_2.

$$E_\text{B} = \left\{\, \{e, r\} \subseteq V_\text{B} \;\middle|\; \exists e' \in \mathcal{E} : (e, r, e') \in \mathcal{D}_\text{KB} \vee (e', r, e) \in \mathcal{D}_\text{KB} \,\right\}$$

This clearly assumes $\mathscr{H}_\text{BICLIQUE}$: for each relation the information of "which e_1" corresponds to "which e_2" is discarded. However this information is not as crucial for entity classification as it is for relation extraction. A small example of graph G_B obtained this way is given in Figure 4.3. The model is trained by jointly optimizing the negative sampling loss and the the log-likelihood of labeled examples. On unseen entities, planetoid reach an accuracy of 61.9% when only 0.1% of entities are labeled.

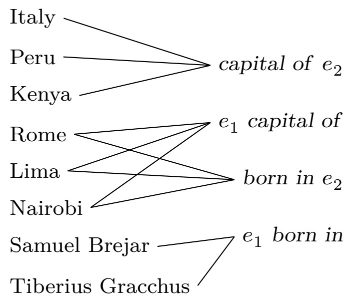


Figure 4.3: NELL dataset bipartite graph. Entities are on the left, while relation slots are on the right. In this graph, the edges are left unlabeled.

Using random walks allows DeepWalk and planetoid to leverage the pre-existing NLP literature. However, for each sample, only a small fraction of the neighborhood—two neighbors at most—of each node is considered to make a prediction. Subsequent methods focused on modeling the information of the whole neighborhood jointly.

4.3.2 Spectral GCN

The first approaches to successfully model the neighborhood of vertices jointly were based on spectral graph theory (Bruna et al. 2014). In practice, this means that the graph is manipulated through its Laplacian matrix instead of directly through the adjacency matrix. In this section, we base our presentation of spectral methods on the work of Kipf and Welling (2017).

Kipf and Welling, "Semi-Supervised Classification with Graph Convolutional Networks" ICLR 2017

We start by introducing some basic concepts from spectral graph theory used to define the convolution operator on graphs. The Laplacian of an undirected graph $G = (V, E)$ can be defined as:

$$\boldsymbol{L}_\text{C} = \boldsymbol{D} - \boldsymbol{M}, \tag{4.1}$$

where $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ is the diagonal matrix of vertex degrees $d_{ii} = \deg(v_i)$ and $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is the adjacency matrix. Equation 4.1 defines the combinatorial Laplacian; however, spectral GCNs are usually defined on the normalized symmetric Laplacian:

$$\boldsymbol{L}_\text{SYM} = \boldsymbol{D}^{-1/2}\boldsymbol{L}_\text{C}\boldsymbol{D}^{-1/2} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{M}\boldsymbol{D}^{-1/2}.$$

Using this definition, we can then take the eigendecomposition of the Laplacian $\boldsymbol{L}_\text{SYM} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{-1}$, where $\boldsymbol{\Lambda}$ is the ordered spectrum—the diagonal matrix of eigenvalues sorted in increasing order—and $\boldsymbol{U}$ is the matrix

The graph Laplacian is similar to the standard Laplacian measuring the divergence of the gradient ($\Delta = \nabla^2$) of scalar functions. Except that the graph gradient is an operator mapping a function on vertices to a function on edges:

$$(\nabla \boldsymbol{f})_{ij} = f_i - f_j$$

And that the graph divergence is an operator mapping a function on edges to a function on vertices:

$$(\operatorname{div} \boldsymbol{G})_i = \sum_{j\in V} m_{ij} g_{ij}$$

Given these definitions, the graph Laplacian is defined as $\Delta = -\operatorname{div}\nabla$. Applying Δ to a signal $\boldsymbol{x} \in \mathbb{R}^n$ is equivalent to multiplying this signal by $\boldsymbol{L}_\text{C}$ as defined in Equation 4.1: $\Delta\boldsymbol{x} = \boldsymbol{L}_\text{C}\boldsymbol{x}$.

of normalized eigenvectors. For an undirected graph, the matrix $\boldsymbol{M}$ is symmetric, therefore $\boldsymbol{U}$ is orthogonal. The orthonormal space formed by the normalized eigenvectors is the Fourier space of the graph. In other words, we can define the graph Fourier transform of a signal $\boldsymbol{x} \in \mathbb{R}^V$ as:

$$\mathscr{F}(\boldsymbol{x}) = \boldsymbol{U}^\mathsf{T}\boldsymbol{x}.$$

Furthermore since the induced space is orthogonal, the inverse Fourier transform is simply defined as:

$$\mathscr{F}^{-1}(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{x}.$$

The expansion of signals in terms of eigenfunctions of the Laplace operator is the leading parallel between the graph Fourier transform and the classical Fourier transform on $\mathbb{R}$ (Shuman et al. 2013). In $\mathbb{R}$, the eigenfunctions $\xi \mapsto e^{2\pi i \xi x}$ correspond to low frequencies when x is small. In the same way, the eigenvectors of the graph Laplacian associated with small eigenvalues assign similar values to neighboring vertices. In particular the eigenvector associated with the eigenvalue 0 is constant with value $1 / \sqrt{n}$. On the other hand, eigenvectors associated with large eigenvalues correspond to high frequencies and encode larger changes of value between neighboring vertices.

Having defined the Fourier transform on graphs, we can use the definition of convolutions as multiplications in the Fourier domain to define convolution on graphs:

$$\boldsymbol{x} * \boldsymbol{w} = \mathscr{F}^{-1}(\mathscr{F}(\boldsymbol{x}) \odot \mathscr{F}(\boldsymbol{w})), \tag{4.2}$$

where $\odot$ denotes the Hadamard (element-wise) product. Note that the convolution operator implicitly depends on the graph G since $\boldsymbol{U}$ is defined from the adjacency matrix $\boldsymbol{M}$. The signal $\boldsymbol{w}$ in Equation 4.2 has the same function as the parametrized filter of CNN (Equation 1.7). Instead of learning $\boldsymbol{w}$ in the spatial domain, we can directly parametrize its Fourier transform $\boldsymbol{w}_{\boldsymbol{\theta}} = \operatorname{diag}(\mathscr{F}(\boldsymbol{w}))$, simplifying Equation 4.2 into:

$\operatorname{diag}(\boldsymbol{x})$ is the diagonal matrix with values of the vector $\boldsymbol{x}$ along its diagonal.

$$\boldsymbol{x} * \boldsymbol{w}_{\boldsymbol{\theta}} = \boldsymbol{U}\boldsymbol{w}_{\boldsymbol{\theta}}\boldsymbol{U}^\mathsf{T}\boldsymbol{x}. \tag{4.3}$$

While $\boldsymbol{w}_{\boldsymbol{\theta}}$ could be learned directly (Bruna et al. 2014), Defferrard et al. (2016) propose to approximate it by Chebyshev polynomials of the first kind (T_k) of the spectrum $\boldsymbol{\Lambda}$:

$$\boldsymbol{w}_{\boldsymbol{\theta}}(\boldsymbol{\Lambda}) = \sum_{k=0}^{K} \theta_k T_k(\boldsymbol{\Lambda}). \tag{4.4}$$

The rationale is that computing the eigendecomposition of the graph Laplacian is too computationally expensive. The Chebyshev polynomials approximation is used to localize the filter; since the k-th Chebyshev polynomial is of degree k, only values of vertices at a distance of at most k are needed.[75] This is similar to how CNNs are usually computed; simple very localized filters are used instead of taking the Fourier transform of the whole input matrix to compute convolution with arbitrarily complex functions. Chebyshev polynomials of the first kind are defined as:

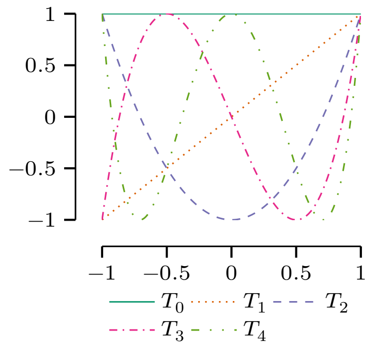
[75] The reasoning behind this localization is the same as the one underlying the fact that the k-th power of the adjacency matrix gives the number of walks of length k (Section 4.2).

$$T_k(\cos x) = \cos(kx). \tag{4.5}$$

Despite its appearance, Equation 4.5 defines a series of polynomials which can be obtained through the application of various trigonometric identities. An alternative but equivalent definition is through the following recursion:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{k+1}(x) &= 2xT_k(x) - T_{k-1}(x) \end{aligned}$$

The plot of the first five Chebyshev polynomials of the first kind follows:



They form a sequence of orthogonal polynomials on the interval $[-1, 1]$ with respect to the weight $1 / \sqrt{1 - x^2}$, meaning that for $k \neq k'$:

$$\int_{-1}^{1} T_k(x) T_{k'}(x) \frac{\mathrm{d}x}{\sqrt{1 - x^2}} = 0.$$

The filter defined by Equation 4.4 is K-localized, meaning that the value of the output signal on a vertex v is computed from the value of $\boldsymbol{x}$ on vertices at distance at most K of v. This can be seen by plugging Equation 4.4 back into Equation 4.3, noticing that it depends on the k-th power of the Laplacian and thus of the adjacency matrix.[76]

Kipf and Welling (2017) proposed to use $K = 1$ with several further optimizations we won't delve into. Using $K = 1$ means that their method computes the activation of a node only from its activation and the activations of its neighbors at the previous layer. This makes the GCN of Kipf and Welling (2017) quite similar to spatial methods described in Section 4.3.3. All the equations given thus far were for a single scalar signal; however, we usually work with vector representations for all nodes, $\boldsymbol{X} \in \mathbb{R}^{n\times d}$. In this case, the layer ℓ of a GCN can be described as:

$$\boldsymbol{H}^{(\ell+1)} = \operatorname{ReLU}\left((\boldsymbol{D}+\boldsymbol{I})^{-1/2}(\boldsymbol{M}+\boldsymbol{I})(\boldsymbol{D}+\boldsymbol{I})^{-1/2}\boldsymbol{H}^{(\ell)}\boldsymbol{\Theta}^{(\ell)}\right)$$

Where $\boldsymbol{\Theta} \in \mathbb{R}^{d\times d}$ is the parameter matrix. Using $\boldsymbol{H}^{(0)} = \boldsymbol{X}$, we can use a GCN with L layers to combine the embeddings in the L-localized neighborhood of each vertex into a contextualized representation.

Kipf and Welling (2017) evaluate their model on the same NELL dataset used by planetoid with the same 0.1% labeling rate. They train their model by maximizing the log-likelihood of labeled examples. They obtain an accuracy of 66.0%, which is an increase of 4.9 points over planetoid.

4.3.3 Spatial GCN

Spatial methods directly draw from the comparison with CNN in the spatial domain. As shown by Figure 4.4, the lattice on which a 2-dimensional[77] CNN is applied can be seen as a graph with a highly regular connectivity pattern. In this section, we introduce spatial GCN by following the GraphSAGE model (Hamilton et al. 2017).

When computing the activation of a specific node with a CNN, the filter is centered on this node, and each neighbor is multiplied with a corresponding filter element. The products are then aggregated by summation. Spatial GCNs purpose to mimic this process. The main obstacle to generalizing this spatial view of convolutions to graphs is the irregularity of neighborhoods.[78] In a graph, nodes have different numbers of neighbors; a fixed-size filter cannot be used. GraphSAGE proposes several aggregators to replace this product–sum process:

Mean aggregator The neighbors are averaged and then multiplied by a single filter $\boldsymbol{W}^{(l)}$:

$$\operatorname{aggregate}^{(\ell+1)}_{\text{mean}}(v) = \sigma\left(\boldsymbol{W}^{(\ell)}\frac{1}{\deg(v)+1}\sum_{u\in N(v)\cup\{v\}}\boldsymbol{h}^{(\ell)}_u\right).$$

A spatial GCN using this aggregator is close to the GCN of Kipf and Welling (2017) with $K = 1$ presented in Section 4.3.2.

LSTM aggregator An LSTM (Section 1.3.2.1) is run through all neighbors with the final hidden state used as the output of the layer.

$$\operatorname{aggregate}^{(\ell+1)}_{\text{LSTM}}(v) = \operatorname{LSTM}^{(\ell)}\left(\left(\boldsymbol{h}^{(\ell)}_u\right)_{u\in N(v)}\right)_{\deg(v)}.$$

Since LSTMs are not permutation-invariant, the order in which the neighbors are presented is important.

Pooling aggregator A linear layer is applied to all neighbors which are then pooled through a max operation.

$$\operatorname{aggregate}^{(\ell+1)}_{\max}(v) = \max\left(\left\{\boldsymbol{W}^{(\ell)}\boldsymbol{h}^{(\ell)}_u + \boldsymbol{b}^{(\ell)} \;\middle|\; u\in N(v)\right\}\right).$$

[76] Derivation of the dependency on $\boldsymbol{L}^k_{\text{SYM}}$ for the proof of K-locality:

$$\begin{aligned}\boldsymbol{x} * \boldsymbol{w}_{\boldsymbol{\theta}}(\boldsymbol{\Lambda}) &= \boldsymbol{U}\left(\sum_{k=0}^{K}\theta_k T_k(\boldsymbol{\Lambda})\right)\boldsymbol{U}^\intercal\boldsymbol{x}\\ &= \left(\sum_{k=0}^{K}\theta_k \boldsymbol{U} T_k(\boldsymbol{\Lambda})\boldsymbol{U}^\intercal\right)\boldsymbol{x}\\ &= \left(\sum_{k=0}^{K}\theta_k T_k(\boldsymbol{L}_{\text{SYM}})\right)\boldsymbol{x}\end{aligned}$$

For the last equality, notice that $\boldsymbol{L}^k_{\text{SYM}} = (\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\intercal)^k = \boldsymbol{U}\boldsymbol{\Lambda}^k\boldsymbol{U}^\intercal$ since $\boldsymbol{U}$ is orthogonal. This can also be applied to the (diagonal) constant term.

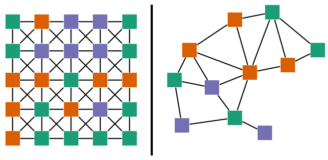


Figure 4.4: Parallel between two-dimensional CNN data and GCN data.

Hamilton et al., "Inductive Representation Learning on Large Graphs" NeurIPS 2017

[77] Even though the same comparison could be made with 1-dimensional CNN as introduced in Section 1.3.1, the similarity is less visually striking. Especially when considering a filter of width 3, in which case the equivalent graph is a simple path graph: .

[78] Interestingly enough, this is also a problem with standard CNNs when dealing with values at the edges of the matrix.

Note that the maximum is applied feature-wise.

Using one of these aggregator, a GraphSAGE layer performs the three following operations for all vertices $v \in V$:

As usual the matrices $\boldsymbol{W}_i^{(l)}$ are trainable model parameters.

$$\begin{aligned}
\boldsymbol{a}_v^{(\ell+1)} &\leftarrow \operatorname{aggregate}^{(\ell+1)}(v) \\
\boldsymbol{h}_v^{(\ell+1)} &\leftarrow \sigma\left(\boldsymbol{W}_1^{(\ell)}\boldsymbol{h}_v^{(\ell)} + \boldsymbol{W}_2^{(\ell)}\boldsymbol{a}_v^{(\ell+1)}\right) \\
\boldsymbol{h}_v^{(\ell+1)} &\leftarrow \boldsymbol{h}_v^{(\ell+1)} \,/\, \left\|\boldsymbol{h}_v^{(\ell+1)}\right\|_2.
\end{aligned}$$

However, this approach still performs poorly when the graph is irregular.[79] In particular, high-degree vertices—such as "United States" in T-REx as described in Section 4.2—incur significant memory usage. To solve this, GraphSAGE proposes to sample a fixed-size neighborhood for each vertex during training. Their representation is therefore computed from a small number of neighbors. Since L layers of GraphSAGE produce L-localized representations, vertices need to be sampled at most at distance L of the vertex for which we want to generate a representation. Hamilton et al. (2017) propose an unsupervised negative sampling loss to train their GCN such that adjacent vertices have similar representations:

[79] In graph theory, a k-regular graph is a graph where all vertices have degree k. By irregular, we mean that the distribution of vertices degrees has high variance; we don't use the term in its formal "highly irregular" meaning. This is indeed the case in scale-free graphs, as their variance is infinite when $\gamma < 3$.

$$\mathcal{L}_{\text{GS}} = \sum_{(u,v)\in E} \log\sigma\left(\boldsymbol{z}_v^\mathsf{T}\boldsymbol{z}_u\right) - \gamma \mathop{\mathbb{E}}_{v'\sim\mathcal{U}(V)}\left[\log\sigma\left(-\boldsymbol{z}_{v'}^\mathsf{T}\boldsymbol{z}_u\right)\right] \tag{4.6}$$

where $\boldsymbol{Z} = \boldsymbol{H}^{(L)}$ is the activation of the last layer and γ is the number of negative samples.

One of the advantages of GraphSAGE compared to the approach of Kipf and Welling (2017) is that it is inductive, whereas the spectral GCN presented in Section 4.3.2 is transductive. Indeed, in the spectral approach, the filter is trained for a specific eigenvectors matrix $\boldsymbol{U}$ which depends on the graph. If the graph changes, everything must be re-trained from scratch. In comparison, the parameters learned by GraphSAGE can be reused for a different graph without any problem.

A limitation of GraphSAGE is that the contribution of each neighbor to the representation of a vertex v is either fixed at $1 \,/\, (\deg(v)+1)$ (with the mean aggregator) or not modeled explicitly. The same can be observed with the model of Kipf and Welling (2017), where the representation of each neighbor u is nonparametrically weighted by $1 \,/\, \sqrt{\deg(v)+\deg(u)}$.

In contrast, graph attention network (GAT, Veličković et al. 2018) proposes to parametrize this weight with a model similar to the attention mechanism presented in Section 1.3.3. The output is built using an attention-like[80] convex combination of transformed neighbors' representations:

Veličković et al., "Graph Attention Networks" ICLR 2018

[80] Veličković et al. (2018) actually propose to use multi-head attention (Section 1.3.4.1). We describe their model with a single attention head for ease of notation.

$$\boldsymbol{h}_v^{(\ell+1)} \leftarrow \sigma\left(\sum_{u\in N(v)\cup\{v\}} \alpha_{vu}^{(\ell)}\boldsymbol{W}^{(\ell)}\boldsymbol{h}_u^{(\ell)}\right),$$

where $\alpha_{vu}^{(\ell)}$, the attention given by v to neighbor u at layer ℓ, is computed using a softmax:

LeakyReLU (Maas et al. 2013) is a variant of ReLU where the negative domain is linear with a small slope instead of being mapped to zero:

$$\operatorname{LeakyReLU}(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise.}\end{cases}$$

$$\alpha_{vu}^{(\ell)} \propto \exp\operatorname{LeakyReLU}\left(\boldsymbol{g}^{(\ell)\mathsf{T}}\begin{bmatrix}\boldsymbol{W}_{\text{GAT}}^{(\ell)}\boldsymbol{h}_v \\ \boldsymbol{W}_{\text{GAT}}^{(\ell)}\boldsymbol{h}_u\end{bmatrix}\right).$$

As usual, the matrices $\boldsymbol{W}$ are parameters, as well as the vector $\boldsymbol{g}$ which is used to combine the representations of the two vertices into a scalar weight.

While GAT and GraphSAGE can be trained in an unsupervised fashion following Equation 4.6, they can also be used as building blocks for larger models, similarly to how we use CNN in Chapter 3. Coupled with the fact that they have a simpler theoretical background and are easier to implement, spatial methods have become ubiquitous to graph-based approaches in the last few years.

4.3.4 GCN on Relation Graphs

All the work introduced in the above sections is about simple undirected graphs $G = (V, E)$. In contrast, in Section 4.1, we encoded the relation extraction problem on attributed multigraphs $G = (\mathcal{E}, \mathcal{A}, \boldsymbol{\varepsilon}, \rho)$. Some works propose to extend GCN to the case of multigraphs, especially when dealing with knowledge bases.[81] This is the case of R-GCN (Schlichtkrull et al. 2018), a graph convolutional network for relational data. The input graph is not labeled with sentences (ς) since R-GCN intents to model a knowledge base $\mathcal{D}_{\text{KB}}$. This means that while G is a multigraph, the subgraphs $G_{\langle r \rangle}$ are simple graphs for all relations $r \in \mathcal{R}$. R-GCNs exploit this by using a separate GCN filter for each relation. An R-GCN layer can be defined as:

$$\boldsymbol{h}_v^{(\ell+1)} \leftarrow \sigma\left(\boldsymbol{W}_0^{(\ell)} \boldsymbol{h}_v^{(\ell)} + \sum_{r\in\mathcal{R}} \sum_{u\in N_{\overrightarrow{\langle r \rangle}}(v)} \boldsymbol{W}_r^{(\ell)} \boldsymbol{h}_u^{(\ell)}\right), \tag{4.7}$$

where $\boldsymbol{W}_0 \in \mathbb{R}^{d'\times d}$ is used for the (implicit) self-loop, while $|\mathcal{R}|$ different filters $\boldsymbol{W}_r \in \mathbb{R}^{d'\times d}$ are used for capturing the arcs. With highly multi-relational data, the number of parameters grow rapidly since a full matrix needs to be estimated for all relations, even rare ones. To address this issue, Schlichtkrull et al. (2018) propose to either constrain the matrices $\boldsymbol{W}_r$ to be block-diagonal, or to decompose them on a small basis $\boldsymbol{Z}^{(\ell)} \in \mathbb{R}^{B\times d'\times d}$:

$$\boldsymbol{W}_r^{(\ell)} = \sum_{b=1}^{B} a_{rb}^{(\ell)} \boldsymbol{Z}_b^{(\ell)},$$

where B is the size of the basis and $\boldsymbol{a}_r$ are the parametric weights for the matrices $\boldsymbol{W}_r$.

[81] In this case, the multigraph is simply labeled since the set of relations is finite. In contrast, in the relation extraction problem, the multigraph is attributed. The arcs are associated with a sentence from an infinite set of possible sentences.

Schlichtkrull et al., "Modeling Relational Data with Graph Convolutional Networks" 2018

Note that only the outgoing neighbors $N_{\overrightarrow{\langle r \rangle}}$ are taken since for each incoming neighbor labeled r, there is an outgoing one labeled $\breve{r}$.

Paralleling the notations used for CNNs in Section 1.3.1, we use d to denote the dimension of embeddings at layer ℓ and d' for the dimension at layer ℓ+1. More often than not, the same dimension is used at all layers $d' = d$. In the following, we use d as a generic notation for embedding and latent dimensions.

Schlichtkrull et al. (2018) evaluate their model on two tasks. First, they evaluate on an entity classification task using a simple softmax layer with a cross-entropy loss on top of the vertex representation at the last layer ($\boldsymbol{H}^{(L)}$ as defined by Equation 4.7). Second, more closely related to relation extraction, they evaluate on a relation prediction task. Given a pair of entity $(e_1, e_2) \in \mathcal{E}^2$, the model must predict the relation $r \in \mathcal{R}$ between them, such that $(e_1, r, e_2) \in \mathcal{D}_{\text{KB}}$. To this end, Schlichtkrull et al. (2018) employ the DistMult model which can be seen as a RESCAL model (Section 1.4.2.2) where the interaction matrices are diagonal. The energy of a fact is defined as:

This is similar to the evaluation of TransE reported in Section 1.4.2.3; except that instead of predicting a missing entity in a tuple $(e_1, r, e_2) \in \mathcal{D}_{\text{KB}}$, the model must predict the missing relation, assuming $\mathscr{H}_{1\text{-ADJACENCY}}$ in the process.

$$\psi_{\text{DistMult}}(e_1, r, e_2) = \boldsymbol{u}_{e_1}^{\mathsf{T}} \boldsymbol{C}_r \boldsymbol{u}_{e_2},$$

where $\boldsymbol{u}_e$ is the embedding of the entity at the last layer of the R-GCN: $\boldsymbol{u}_e = \boldsymbol{h}_e^{(L)}$ and $\boldsymbol{C}_r \in \operatorname{diag}(\mathbb{R}^d)$ is a diagonal matrix parameter. The probability associated to a fact by DistMult is proportional to the exponential of the energy function ψ_{DistMult}. Therefore, a missing relation between $e_1, e_2 \in \mathcal{E}$ can be predicted by taking the softmax over relations $r \in \mathcal{R}$

of $\psi_{\text{DistMult}}(e_1, r, e_2)$. R-GCNs are trained using negative sampling (Section 1.2.1.3) on the entity classification and relation prediction tasks. This is similar to the training of TransE, where the main difference is that the entity embeddings are computed using R-GCN layers instead of being directly fetched from an entity embedding matrix.

A limitation of R-GCNs is that they only rely on vertices' representation. Even when the evaluation involves the classification of arcs (as is the case with relation prediction), this is only done by combining the representations of the endpoints (using DistMult).

Several works build upon R-GCN. GP-GNN (H. Zhu et al. 2019) applies a similar model to the supervised relation extraction task. In this case, the graph is attributed with sentences instead of relations; therefore, the weight matrices $\boldsymbol{W}_r$ are generated from the sentences instead of using an index of all possible relations. They apply their model to Wikipedia distantly supervised by Wikidata. However, the classification is still made from the representation of the endpoints of arcs. Related work also appears in the *heterogeneous graph* community (Z. Hu et al. 2020; X. Wang et al. 2019). Heterogeneous graphs are graphs with labels on both vertices and arcs. The model proposed by Z. Hu et al. (2020) is similar to R-GCN with an attention mechanism more akin to the transformer's attention (Section 1.3.4.1) than classical attention (Section 1.3.3). The canonical evaluation datasets of this community are citation graphs. Vertices are assigned labels such as "people," "article" and "conference," while arcs are labeled with a small number of domain-specific relations: *author*, *published at*, *cite*, etc. The evaluation task typically corresponds to entity prediction.

H. Zhu et al., "Graph Neural Networks with Generated Parameters for Relation Extraction" ACL 2019

Z. Hu et al., "Heterogeneous Graph Transformer" WWW 2020

X. Wang et al., "Heterogeneous Graph Attention Network" WWW 2019

4.3.5 Weisfeiler–Leman Isomorphism Test

In this section, we introduce the theoretical background of GCNs. This is of particular interest to us since this theoretical background is more closely related to unsupervised relation extraction than GCNs can be at first glance. As stated in the introduction to the thesis, relations emerge from repetitions. In particular, we expect that two identical (sub-)graphs convey the same relations. However, testing whether two graphs are identical is a complex problem. Indeed, we have to match each of the n vertices of the first graph to one of the n possibilities in the second graph. Naively, we need to try all $n!$ possibilities. This is known as the graph isomorphism problem. Two simple graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ are said to be isomorphic ($G_1 \simeq G_2$) iff there exists a bijection $f\colon V_1 \to V_2$ such that $(u, v) \in E_1 \iff (f(u), f(v)) \in E_2$. Figure 4.5 gives an example of two isomorphic graphs.

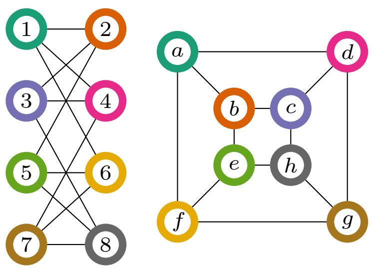


Figure 4.5: Example of isomorphic graphs. Each vertex i in the first graph corresponds to the i-th letter of the alphabet in the second graph. Alternatively, these graphs have nontrivial automorphism, for example, by mapping vertex i to vertex $9 - i$.

The various GCN methods introduced thus far can be seen as generalizations of the Weisfeiler–Leman[82] isomorphism test (Weisfeiler and Leman 1968), which tests whether two graphs are isomorphic. The k-dimensional Weisfeiler–Leman isomorphism test (k-dim WL) is a polynomial-time algorithm assigning a color to each k-tuple of vertices[83] such that two isomorphic graphs have the same coloring. With a bit of work, the general k-dim WL algorithm can be implemented in $O(k^2 n^{k+1} \log n)$ (Immerman and Lander 1990). However, there exist pairs of graphs that are not isomorphic, yet are assigned with the same coloring by the Weisfeiler–Leman test (Cai et al. 1992). At the time of writing, the precise membership of the graph isomorphism problem with respect to the polynomial complexity classes is still conjectural. No polynomial-time algorithm nor reduction

[82] Often spelled Weisfeiler–Lehman, Babai (2016) indicates that Andreĭ Leman preferred to transliterate his name without an "h."

Weisfeiler and Leman, "The reduction of a graph to canonical form and the algebra which appears therein" NTI 1968

[83] An ordered sequence of k vertices, that is an element of V^k, not necessarily connected.

Cai et al., "An optimal lower bound on the number of variables for graph identification" Combinatorica 1992

algorithm Weisfeiler–Leman
  Inputs: $G = (V, E)$ graph
    k dimensionality
  Output: χ_∞ coloring of k-tuples

  ▷ *Initialization* ◁
  $\ell \leftarrow 0$
  for all $\boldsymbol{x} \in V^k$ **do**
    $\chi_0(\boldsymbol{x}) \leftarrow \mathrm{iso}(\boldsymbol{x})$
  ▷ *Main Loop* ◁
  repeat
    $\ell \leftarrow \ell + 1$
    $\mathfrak{I}_\ell \leftarrow$ new color index
    for all $\boldsymbol{x} \in V^k$ **do**
      $c_\ell(\boldsymbol{x}) \leftarrow \{\!\{ \chi_{\ell-1}(\boldsymbol{y}) \mid \boldsymbol{y} \in N^k(\boldsymbol{x}) \}\!\}$
      $\chi_\ell(\boldsymbol{x}) \leftarrow$ index of $(\chi_{\ell-1}(\boldsymbol{x}), c_\ell(\boldsymbol{x}))$ in $\mathfrak{I}_\ell$
  until $\chi_\ell = \chi_{\ell-1}$
  output χ_ℓ

Algorithm 4.2: The Weisfeiler–Leman isomorphism test. The double braces $\{\!\{\ \}\!\}$ denote a multiset. Since $\mathfrak{I}_\ell$ is indexed with the previous coloring $\chi_{\ell-1}(\boldsymbol{x})$ of the vertices—alongside $c_\ell(x)$—the number of color classes is strictly increasing until the last iteration when it remains constant. Since the last coloring is stable, we refer to it as χ_∞.

from NP-complete problems are known. This makes graph isomorphism one of the prime candidates for the NP-intermediate complexity class.[84]

The general k-dim WL test is detailed in Algorithm 4.2. It is a refinement algorithm, which means that at a given iteration, color classes can be split, but two k-tuples with different colors at iteration ℓ can't have the same color at iteration $\ell' > \ell$. Initially, all k-tuples x are assigned a color according to their isomorphism class $\mathrm{iso}(x)$. We define the isomorphism class through an equivalence relation. For two k-tuples $\boldsymbol{x}, \boldsymbol{y} \in V^k$, $\mathrm{iso}(x) = \mathrm{iso}(y)$ iff:[85]

- $\forall i, j \in [1, \dots, k] : x_i = x_j \iff y_i = y_j$
- $\forall i, j \in [1, \dots, k] : (x_i, x_j) \in E \iff (y_i, y_j) \in E$

Intuitively, this checks whether $x_i \mapsto y_i$ is an isomorphism for the subgraphs built from the k vertices $\boldsymbol{x}$ and $\boldsymbol{y}$. This is not the same as the graph isomorphism problem since here, the candidate isomorphism is given, we don't have to test the $k!$ possibilities.

The coloring of $\boldsymbol{x} \in V^k$ is refined at each step by juxtaposing it with the coloring of its neighbors $N^k(\boldsymbol{x})$. We need to reindex the new colors at each step since the length of the color strings would grow exponentially otherwise. The set of neighbors[86] of a k-tuple for $k \geq 2$ is defined as:

$$N^k(\boldsymbol{x}) = \left\{ \boldsymbol{y} \in V^k \;\middle|\; \exists i \in [1, \dots, k] : \forall j \in [1, \dots, k] : j \neq i \implies x_j = y_j \right\}.$$

In other words, the k-tuples $\boldsymbol{y}$ neighboring $\boldsymbol{x}$ are those differing by at most one vertex with $\boldsymbol{x}$.

The 1-dim WL test is also called the *color refinement* algorithm. In this case, $N^1(x)$ is simply $N(x)$ the set of neighbors of x. The isomorphism class of a single vertex is always the same, so χ_0 assigns the same color to all vertices. The first iteration of the algorithm groups vertices according to their degree (the multiplicity of the sole element in the multiset $c_1(x)$). The second iteration χ_2 then colors each vertex according to its degree χ_1 and the degree of its neighbors c_2. And so on and so forth until χ does not change anymore.

[84] The class of NP problems neither in P nor NP-complete. It is guaranteed to be non-empty if P $\neq$ NP. Clues for the NP-intermediateness of the graph isomorphism problem can be found in the fact that the counting problem is in NP (Mathon 1979) and more recently, from the fact that a quasi-polynomial algorithm exists (Babai 2015).

[85] To avoid having to align two colorings, the WL algorithm is usually run on the disjoint union of the two graphs. So, strictly speaking, it tests for automorphism (isomorphic endomorphism). Therefore we can assume $\boldsymbol{x}$ and $\boldsymbol{y}$ are from the same vertex set V.

[86] Note that the kind of neighborhood defined by N^k completely disregards the edges in the graph. For this reason, it is sometimes called the *global neighborhood*.

The GCN introduced in the previous sections can be seen as variants of the 1-dim WL algorithm where the index $\mathfrak{I}_\ell$ is replaced with a neural network such as $\text{aggregate}^{(\ell)}_{\text{mean}}$ given in Section 4.3.3. In this case χ_ℓ corresponds to $\boldsymbol{H}^{(\ell)}$ the activations at layer ℓ.

4.4 Proposed Approaches

We now turn to the graph-based models we propose to leverage information from the structure of the dataset. Let us quickly summarize the context in which we inscribe our work. We have access to two kinds of features: linguistic—from the sentence—and topological—from the graph. Unsupervised relation extraction methods do not fully exploit graph neighborhoods.[87] Supervised methods such as EPGNN and GP-GNN do, even though the information present in the graph is more important in the unsupervised setting. Indeed, the relational information is mostly extractable from the sentences and entities alone. While extra information from topological features can still be used by supervised models, it is not essential. On the other hand, in the unsupervised setting, the main issue is to identify the relational information in the sentence, to distinguish it from other semantic contents. As we show in Section 4.2, this relational information is also present in the topological features (the neighborhood of a sample). This can be useful in two ways:

[87] As explained in Section 4.1, MTB does use close neighborhoods as contrast during training, but not for inference.

1. Use both pieces of information jointly, linguistic and topological: "the more features, the better." This is what supervised models do.
2. Use the topological features to identify the relational information in the linguistic features.

In Section 4.4.1, we exploit the first point by adding a GCN to the matching the blanks model (MTB, Section 2.5.6). In Section 4.4.2, we show that topological features can be used without training a GCN. This also serves as an introduction to Section 4.4.3, which proposes an unsupervised loss following the second point above; it exploits the fact that relation information is present in both linguistic and topological features.

4.4.1 Using Topological Features

In this section, we seek to use topological information as additional features for an existing unsupervised model: matching the blanks (MTB). The usefulness of these features lies in the fact that many relations are "typed": e.g. they only accept geographical locations as objects and only people as subjects (such as *born in*). This can be captured by looking at the neighborhood of each entity, which can be seen as a "soft" version of $\mathscr{H}_{\text{TYPE}}$ ("relations are typed," Section 2.5.3).

A straightforward approach is to parallel the construction of R-GCN (Section 4.3.4): use a GCN-like encoder followed by a relation classifier—in the case of R-GCN, DistMult. In effect, this corresponds to taking MTB and augmenting it with a GCN to process neighboring samples. As a reminder, MTB uses a similarity-based loss where each unsupervised sample $(s, \boldsymbol{e}) \in \mathcal{D}$ is represented by $\text{BERTcoder}(s)$. In this model, the information lies on the arcs. In order to use a GCN model, we transform our graph $G = (\mathcal{E}, \mathcal{A}, \varepsilon, \rho, \varsigma)$ such that the information lies on the vertices instead.

This transformed graph is called the *line graph* of G and noted $L(G)$. An illustration for simple undirected graphs is provided in Figure 4.6. For a directed (multi)graph, it is defined as follows:

$$\begin{aligned} L(G) &= (\mathcal{A}, \mathfrak{A}, \boldsymbol{\varepsilon}, \varsigma) \\ \mathfrak{A} &= \left\{\, (a_1, a_2) \in \mathcal{A}^2 \;\middle|\; \varepsilon_2(a_1) = \varepsilon_1(a_2) \,\right\}. \end{aligned}$$

In other words, each arc becomes a vertex and an arc $a_1 \to a_2$ is present if and only if a_1 and a_2 form a directed path of length 2. The neighborhood of each sample (arc is the original G) is still defined as all other samples with at least one entity in common since by construction for all V-structures $e_1 \xrightarrow{a_1} e_2 \xleftarrow{a_2} e_3$, there exists a directed path $e_1 \xrightarrow{a_1} e_2 \xrightarrow{\breve{a}_2} e_3$ in the original graph G. This construction is actually similar to the one of EPGNN introduced in Section 2.4.5. The main difference is that each vertex in $L(G)$ corresponds to a sample in $\mathcal{D}$, while an EPGNN graph groups samples by entity pairs into a single vertex.

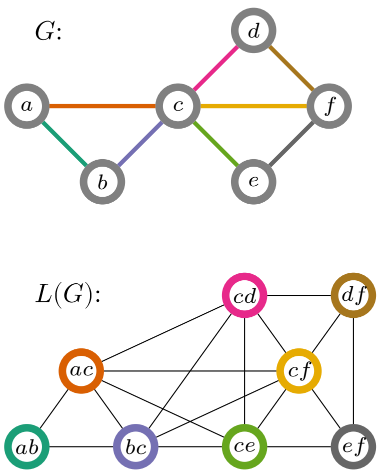


Figure 4.6: Example of line graph construction. Each edge $x — y$ in the simple undirected graph G corresponds to the vertex xy with the same color in the graph $L(G)$. Two vertices in $L(G)$ are connected iff the corresponding edges share an endpoint in G. In directed graphs, the two arcs further need to be in the same direction in G for an arc to exist in $L(G)$.

The standard loss and training algorithm of MTB as defined by Equation 2.10 can be reused as is, we only need to redefine the similarity function (Equation 2.9):

$$\operatorname{sim}(a, a', G) = \sigma\left(\begin{aligned} &\operatorname{BERTcoder}(\varsigma(a))^\mathsf{T} \operatorname{BERTcoder}(\varsigma(a')) \\ &\qquad + \lambda \operatorname{GCN}(L(G))_a^\mathsf{T} \operatorname{GCN}(L(G))_{a'} \end{aligned}\right), \tag{4.8}$$

where λ is a hyperparameter weighting the topological-based prediction over the sentence-based one. At the input of the GCN, the vertices are labeled using the same sentence encoder: $\boldsymbol{x}_a = \operatorname{BERTcoder}(\varsigma(a))$.

The only difference between MTB and the MTB–GCN hybrid we propose is the additional λ-weighted term in Equation 4.8. We use this model to evaluate whether topological features can be exploited by an existing unsupervised relation extraction loss. It tells us how much can be gained from the "adding more features" aspect of graph-based methods and contrast it with the new topology-aware loss design we propose in Section 4.4.3.

4.4.2 Nonparametric Weisfeiler–Leman Iterations

The losses used to train unsupervised GNNs usually make the hypothesis that linked vertices should have similar representations. This can be seen in $\mathcal{L}_{\text{GS}}$ (Equation 4.6), which seeks to maximize the dot product between the representations of adjacent vertices. While this hypothesis might be helpful for most problems on which GNNs are applied, this is clearly not the case for relation extraction. In Section 4.4.1, we introduced a first simple solution to this problem is to replace the loss used by the GNN with a standard unsupervised relation extraction loss. However, it is also possible to design an unsupervised loss from the theoretical foundation of GCN: the Weisfeiler–Leman isomorphism test. To this end, we propose to build a model relying on the following hypothesis:

Weak Distributional Hypothesis on Relation Extraction Graph:
Two arcs conveying similar relations have similar neighborhoods.

Note that we dubbed this version of the distributional hypothesis *weak* since we only state it in one direction, the converse having several counterexamples. For example, sentences about the place of birth and the place of death of a person tend to have similar neighborhoods despite conveying different relations.[88] To distinguish these kinds of relations with similar neighborhoods, we have to rely on sentence representations.[89]

Following this hypothesis, we first propose a simple parameter-less approach based on the Weisfeiler–Leman isomorphism test (Section 4.3.5). *We can say that two neighborhoods are similar if they are isomorphic.* Therefore, we can enforce the hypothesis above by ensuring that if two neighborhoods are assigned similar coloring by the WL algorithm, they convey similar relations. In the relation extraction problem, contrary to much of the related work presented in Section 4.3, we have data on the arcs of the graph, not on the vertices. This means that instead of using the 1-dimensional Weisfeiler–Leman algorithm, we use the 2-dimensional version. In other words, instead of coloring the vertices, we color the arcs since our problem is to label them with a relation.

The initial coloring $\chi_0(a)$ is initialized as the isomorphism class of a sample $a \in \mathcal{A}$. We can define this isomorphism class using BERTcoder(a), which means that the initial representation of a sample will simply be the sentential representation of the sample. The difficult task is to define the re-indexing of colors as performed by $\mathfrak{I}$ in Algorithm 4.2. This is difficult since the original WL algorithm is defined on a discrete set of colors, while we need to manipulate distributed representations of sentences.

If we want to produce clear-cut relation classes, we can use a hashing algorithm on sentence representations such as the one proposed for graph kernels by Morris et al. (2016). However, we focus on a few-shot evaluation in order to compare with MTB and to avoid errors related to knowledge base design as described in Section 2.5.1.2. In this case, we only need to be able to compare the colors of two different samples, measuring how close they are to each other. Let us define $\mathcal{N} : \mathcal{A} \to 2^{\mathcal{A}}$ the function mapping an arc to the set of its neighbors. Formally, for $a \in \mathcal{A}$, $\mathcal{N}(a) = \{a' \in \mathcal{A} \mid \boldsymbol{\varepsilon}(a) \cap \boldsymbol{\varepsilon}(a') \neq \emptyset\}$. In other words, $\mathcal{N}$ in G corresponds to the neighbors function N in the line graph $L(G)$. Since $\mathcal{A}$ can be seen as the set of samples, $\mathcal{N}(a)$ can be seen as the set of samples with at least one entity in common with a. To enforce the weak distributional hypothesis on graphs stated above, we take two first-order neighborhoods $\mathcal{N}(a), \mathcal{N}(a') \subseteq \mathcal{A}$ and define a distance between them. This corresponds to comparing two empirical distributions of sentence representations[90] that have an entity in common with a and a'. This can be done using the 1-Wasserstein distance between the two neighborhoods since they can be seen as two distributions of Dirac deltas in BERTcoder representation space.[91] This needs to be done for the two entities, which correspond to the in-arc-neighbors $\mathcal{N}_{\leftarrow}$ and out-arc-neighbors $\mathcal{N}_{\rightarrow}$. While this is 1-localized, we can generalize this encoding to be K-localized by defining the k-sphere centered on an arc a, where the 1-sphere corresponds to $\mathcal{N}$:

$$\begin{aligned} S_{\rightarrow}(a, 0) &= \{\, a \,\} \\ S_{\rightarrow}(a, k) &= \{\, x \in \mathcal{A} \mid \exists y \in S_{\rightarrow}(a, k-1) : \varepsilon_1(x) = \varepsilon_2(y) \,\}. \end{aligned}$$

This sphere can be embedded using BERTcoder, which corresponds to retrieving its initial coloring:

$$\mathfrak{S}_{\rightarrow}(a, k) = \{\, \text{BERTcoder}(\varsigma(x)) \in \mathbb{R}^d \mid x \in S_{\rightarrow}(a, k) \,\}.$$

We can thereafter define the K-localized out-neighborhood of $a \in \mathcal{A}$ as the sequence of $\mathfrak{S}_{\rightarrow}(a, k)$ for all $k = 1, \ldots, K$. The in-neighborhood is defined

[88] The neighborhoods are somewhat dissimilar in that "notable" people tend to die in places with more population than their birthplace. However, whether current models can pick this up from other kinds of regularity in a dataset is dubious.

[89] This can partly explain the conditional entropy $\mathrm{H}(\mathrm{r}_2 \mid \mathrm{r}_1, \mathrm{r}_3) \approx 1.06$ bits given in Section 4.2.

The astute reader might have noticed that the 2-dimensional WL isomorphism test as described in Algorithm 4.2 loops over pairs of vertices, not arcs. This is impractical in our relation extraction graph, which is particularly sparse—the number of arcs m is far larger than the number of vertices n. The extra (unlinked) entity pairs considered by Algorithm 4.2 are usually referred to as *anti-arcs*. Ignoring anti-arcs leads to the local Weisfeiler–Leman isomorphism tests since only the "local neighborhood" is considered. Other intermediate approaches are possible, sometimes referred to as the *glocalized* variants of Weisfeiler–Leman. See Morris et al. (2020) for an example of application to graph embeddings. Alternatively, our proposed approach can be seen as a 1-dimensional Weisfeiler–Leman isomorphism test applied to the line graph.

[90] We are comparing sentence representations and not directly sentences since the initial coloring χ_0 has been defined using BERTcoder.

[91] Wasserstein distance has the advantage of working on distributions with disjoint supports.

similarly. Finally, the distance between two samples $a, a' \in \mathcal{A}$ can be defined as:

$$d(a, a'; \boldsymbol{\lambda}) = \sum_{k=0}^{K} \frac{\lambda_k}{2} \sum_{o \in \{\leftarrow, \rightarrow\}} W_1\left(\mathfrak{S}_o(a, k), \mathfrak{S}_o(a', k)\right), \tag{4.9}$$

To be precise Equation 4.9 defines a distance between samples from the Euclidean distances between neighboring samples—that is samples with an entity in common. The distance W_1 is the cost of the optimal transport plan between two sets of Dirac deltas corresponding to the neighborhoods of the samples.

where W_1 designates the 1-Wasserstein distance, and $\boldsymbol{\lambda} \in \mathbb{R}^{K+1}$ weights the contribution of each sphere to the final distance value. In particular λ_0 parametrizes how much the linguistic features should weight compared to topological features.[92]

[92] The 1-Wasserstein distance is defined on top of a metric space; therefore, the difference between two neighbors must be defined using the Euclidean distance. We can't use dot product as usually done with BERT representations (see for example Equation 2.9). However, we can slightly change Equation 4.9 to use the dot product for the computation of the linguistic similarity (the term $k = 0$). In this case, however, d would no longer satisfy the properties of a metric.

To relate this function back to our original re-coloring problem, the distance d up to K can be seen as a distance on χ_K, the coloring assigned at step K. Indeed, if $d(a, a', \boldsymbol{\lambda}) = 0$ then $\chi_K(a) = \chi_K(a')$. However, while two colors are either equal or not in the original algorithm, the distance d gives a topology to the set of arcs. We don't directly compute a hard-coloring of 2-tuples. The closest thing to a coloring χ in our algorithm is the sphere embedding $\mathfrak{S}$, which in fact, is more akin to c in Algorithm 4.2. In other words, we skip the re-indexing step of the Weisfeiler–Leman algorithm to deal with the continuous nature of sentence embeddings at the cost of a higher computational cost.

Combining a Wasserstein distance with Weisfeiler–Leman was already proposed for graph kernels (Togninalli et al. 2019). However, this was applied to a simple graph without attributed edges, and it was unrelated to any information extraction task. For unsupervised relation extraction, the distance function d can directly be used to compute the similarity between query and candidates samples in a few-shot problem (Section 2.5.1.2). Since the number of arcs at distance k grows quickly in a scale-free graph,[93] we either need to keep K low or employ sampling strategies similarly to GraphSAGE (Section 4.3.3). Furthermore, the Wasserstein distance is hard to compute exactly; entropic regularization of the objective has been proposed. In particular, W_1 can be efficiently computed with Sinkhorn iterations (Cuturi 2013).

[93] Remember that the diameter of the (scale-free) graph is in the order of $\log \log n$.

4.4.3 Refining Linguistic and Topological Features

While the nonparametric method presented in the previous section manages to consider both the linguistic and topological features, it processes them in isolation. In this section, we propose a scheme that allows both the encoder of linguistic and topological features to adapt to each other in a training process. Conceptually, this is somewhat similar to SelfORE (Section 2.5.7). As a reminder, SelfORE is a clustering method that purifies relation clusters by optimizing BERTcoder such that samples with close linguistic forms are pushed closer. In our scheme, we propose to refine both linguistic and topological features with respect to each other. In this way we hope to both enforce $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$ and the following assumption:

As a reminder, $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$ states that two samples with similar contextualized embeddings convey similar relations. See Appendix B.

Assumption $\mathscr{H}_{\text{1-NEIGHBORHOOD}}$: *Two samples with the same neighborhood in the relation extraction graph convey the same relation.*
$\forall a, a' \in \mathcal{A} \colon \mathcal{N}(a) = \mathcal{N}(a') \implies \rho(a) = \rho(a')$

Note that this is the converse of the weak distributional hypothesis on relation extraction graph stated in Section 4.4.2. We need to make the modeling hypothesis in this direction since in the unsupervised relation extraction problem, we do not have access to relations and therefore can't enforce an hypothesis between samples conveying the same relations. We posit that by balancing $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$ and $\mathscr{H}_{\text{1-NEIGHBORHOOD}}$ we are able to exploit the structure induced by both sources information in an unsupervised samples $(s, \boldsymbol{e}) \in \mathcal{D}$: the sentence s and entities $\boldsymbol{e}$, whereas SelfORE only relies on the sentence s.

To define the topological and linguistic distance between two samples, we use the distance function defined by Equation 4.9. For computational reasons, we set $K = 1$, which means that our model is 1-localized. The linguistic distance is simply the distance between the BERTcoder of the samples' sentences. In other words, it is $d(a, a'; [1, 0]^\mathsf{T})$. On the other hand, the topological distance can be defined as the distance between the two neighborhoods, in other words, $d(a, a'; [0, 1]^\mathsf{T})$. We propose to train BERTcoder such that these two distances coincide more. In practice, this can be achieved with a triplet loss similar to the one used by TransE (Section 1.4.2.3). Given three arcs $\boldsymbol{a} \in \mathcal{A}^3$, we ensure the two distances are similar between the two first arcs a_1 and a_2, and we contrast these distances using the third arc a_3. This translates to the following loss:

Intuitively, we want to optimize the mean squared error (MSE) between the linguistic and topological features of all pairs of arcs $(d(a_1, a_2, [1, 0]^\mathsf{T}) - d(a_1, a_2, [0, 1]^\mathsf{T}))^2$. However, this loss could be optimized by encoding all arcs into a single point. The output of BERTcoder would then be constant. Therefore, we need to regularize the MSE loss such that distances that shouldn't be close are not. This is the point of the triplet loss; we contrast the positive distance delta with a negative one. While $d(a_1, a_2, [1, 0]^\mathsf{T})$ and $d(a_1, a_2, [0, 1]^\mathsf{T})$ should be close to each other (because of $\mathscr{H}_{\text{1-NEIGHBORHOOD}}$), they shouldn't be close to any distance involving a third sample a_3. This ensures that our model does not collapse.

$$\mathcal{L}_{\text{LT}}(a_1, a_2, a_3) = \max \begin{pmatrix} 0, \zeta + 2\big(d(a_1, a_2, [1,0]^\mathsf{T}) - d(a_1, a_2, [0,1]^\mathsf{T})\big)^2 \\ \qquad - \big(d(a_1, a_2, [1,0]^\mathsf{T}) - d(a_1, a_3, [0,1]^\mathsf{T})\big)^2 \\ \qquad - \big(d(a_1, a_3, [1,0]^\mathsf{T}) - d(a_1, a_2, [0,1]^\mathsf{T})\big)^2 \end{pmatrix},$$

where $\zeta > 0$ is a hyperparameter defining the maximum margin we seek to enforce between the true distance-error and the negative distance-error. By randomly sampling arcs triplets $\boldsymbol{a} \in \mathcal{A}^3$, we can fine-tune a BERTcoder in an unsupervised fashion such that it captures both linguistics and topological features. During evaluation, the procedure described in Section 4.4.2 can be reused, such that both linguistic representations refined by the topological structure and the topological representations refined by the linguistic structure are used jointly. However, both distances could be used independently, for example if a sample contains unseen entities, or on the contrary if we want to assess which relation links two entities without any supporting sentence.

4.5 Experiments

Matching the blanks was trained on a huge unsupervised dataset that is not publicly available (Soares et al. 2019). To ensure reproducibility, we instead attempt to train on T-REx (Section C.7, Elsahar et al. 2018). The evaluation is done in the few-shot setting (Section 2.5.1.2) on the FewRel dataset (Section C.2) in the 5-way 1-shot setup. Our code is available at `https://esimon.eu/repos/gbure`.

Elsahar et al., "T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples" LREC 2018

The BERTcoder model we use is the entity markers–entity start described in Section 2.3.7, based on a `bert-base-cased` transformer. We use a BERTcoder with no post-processing layer for the standalone BERT model. The MTB model is followed by a layer norm even during pre-training as described by Soares et al. (2019). The MTB similarity function remains a dot product but was rescaled to be normally distributed. When augmenting MTB with a GCN, we tried both the Chebyshev approximation described in Section 4.3.2 and the mean aggregator of Section 4.3.3, however we were only able to train de Chebyshev variant at the time of writing. The nonparametric WL algorithm uses a dot product for linguistic similarity and a Euclidean 1-Wasserstein distance for topological distance; the hyperparameters are $\boldsymbol{\lambda} = [-1, 0.2]^\mathsf{T}$.

We report our results in Table 4.2. The given numbers are accuracies on the subset of FewRel with at least one neighbor in T-REx. The accuracies on the whole dataset are 73.74% for linguistic features alone (BERT) and 77.54% for MTB. Our results for MTB are still slightly below what Soares et al. (2019) report because of the BERT model size mismatch and the smaller pre-training dataset. The result gap is within expectations, as already reported by other works that used a similar setup on the supervised setup (Qu et al. 2020). On the other hand, our accuracy for a standalone BERT is higher than what Soares et al. (2019) report; we suspect this is due to our removal of the randomly initialized post-processing layer.

Model	Accuracy
Linguistic (BERT)	69.46
Topological (W_1)	65.75
Nonparametric WL	72.18
MTB	78.83
MTB GCN–Chebyshev	76.10

Table 4.2: Preliminary results for FewRel valid accuracies of graph-based approaches. To better evaluate the efficiency of topological features, we report results on the subset of the dataset that is connected in T-REx.

The top half of Table 4.2 reports results for nonparametric models. These models were not trained for the relation extraction task; they simply exploit an MLM-pretrained BERT in clever ways. As we can see, while topological features are a bit less expressive to extract relations by themselves, they still contain additional information that can be used jointly with linguistic features—this is what the nonparametric WL model does.

For parametric models, we have difficulties training on T-REx because of its relative small size. In practice 66.89% of FewRel entities are already mentioned in T-REx. However, a standard 5-way 1-shot problem contains $(1 + 5) \times 2 = 12$ different entities. We measure the empirical probability that all entities of a few-shot problem are connected in T-REx to be around 0.54%. Furthermore, we observe that MTB augmented with a GCN performs worse than a standalone MTB despite adding a single linear layer to the parameters (the BERTcoder of the linguistic and topological distances are shared). These are still preliminary results, however, it seems the small size of T-REx coupled with the large amount of additional information presented to the model cause it to overfit on the train data. We observe a similar problem with the triplet loss model of Section 4.4.3. At the time of writing, our current plan is to attempt training on a larger graph, similar to the unsupervised dataset of Soares et al. (2019).

4.6 Conclusion

In this chapter, we explore aggregate approaches to unsupervised relation extraction using graphs. In Section 4.2, we show that a large amount of information can be leveraged from the neighborhood of a sample. This, together with the observation that previous unsupervised methods always ignored the neighborhood of a sample at inference, opens a new research direction for unsupervised methods. In Section 4.4, we propose several models that make use of the neighborhood information. In particular, we propose a novel unsupervised training loss in Section 4.4.3, which makes very few modeling assumptions while still being able to exploit the neighborhood information both at training and prediction time.

Our contributions lie in using a multigraph with arcs attributed with sentences (Sections 4.1), our method to approximate the quantity of information extractible from this graph (Sections 4.2) and our proposed approach to utilize this additional information (Section 4.4). Despite encouraging early results showing the soundness of using the relation extraction graph, at the present time we only improved nonparametric models. More experimentation is still needed to fully exploit topological information.

Conclusion

During this Ph.D. candidacy, I—mostly[94]—focused on the study of unsupervised relation extraction. In this task, given a set of tagged sentences and pairs of entities, we seek the set of conveyed facts (e_1, r, e_2), such that r embodies the relationship between e_1 and e_2 expressed in some sample. To tackle this task, we follow two main axes of research: first, the question of how to train a deep neural network for unsupervised relation extraction; second, the question of how to leverage the structure of an unsupervised dataset to gain additional information for the relation extraction task.

[94] With the occasional—and deeply appreciated—distraction of Syrielle Montariol on unrelated NLP projects (Montariol et al. 2022).

Summary of Contributions

For more than a decade now, the field of machine learning has been overrun by deep learning approaches. Since I started working on unsupervised relation extraction in late 2017, the task followed the same fate. The VAE model of Marcheggiani and Titov (2016) started introducing deep learning methods to the task. However, it was still limited by a sentence representation based on hand-engineered features. My first axis of research was to partake in this deep learning transition (Chapter 3). Subsequently, the use of deep learning was made simpler with the replacement of CNN and LSTM-based models with pre-trained transformers. Indeed, a model like BERT (Devlin et al. 2019) performs reasonably well on unsupervised relation extraction "out of the box." This was exploited by others, in the clustering setup by SelfORE (X. Hu et al. 2020), and in the few-shot setup by MTB (Soares et al. 2019). My second axis of research was to exploit the regularities of the dataset to leverage additional information from its structure (Chapter 4). While some works already used this information in supervised relation extraction (Chen et al. 2006; Zhao et al. 2019), unsupervised models made no attempt at modeling it explicitly. Our proposed approaches are based on a graph representation of the dataset. As we have shown, they inscribe themselves in a general revival of graph-based approaches in deep learning (Hamilton et al. 2017; Kipf and Welling 2017). We now describe the three main contributions we can draw from our work.

Marcheggiani and Titov, "Discrete-State Variational Autoencoders for Joint Discovery and Factorization of Relations" TACL 2016

X. Hu et al., "SelfORE: Self-supervised Relational Feature Learning for Open Relation Extraction" EMNLP 2020

Soares et al., "Matching the Blanks: Distributional Similarity for Relation Learning" ACL 2019

Literature review with formalized modeling assumptions.
In Chapter 2, we presented relevant relation extraction models from the late 1990s until today. We first introduced supervised approaches, which we split into two main blocks:

Sentential methods extract a relation for each sample in isolation. In this setup, there is no difference between evaluating a model on a single dataset with a thousand samples or a thousand datasets containing one sample each. Indeed, these models do not model the interactions between samples.

Aggregate methods map a set of unsupervised samples to a set of facts at once. There is not necessarily a direct correspondence between extracted facts and samples in the dataset, even though most aggregate models still provide a sentential prediction. In this setup, a dataset containing a single sentence would be meaningless; it would boil down to a sentential approach.

This distinction can also be made for unsupervised models, and indeed Chapter 3 follows mostly a sentential approach, whereas Chapter 4 purposes to introduce the aggregate approach to the unsupervised setting.

In Chapter 2, we also presented unsupervised relation extraction models. Unsupervised models need to rely on modeling hypotheses to capture the notion of relation. While these hypotheses are not always clearly stated in articles, they are central to the design of unsupervised approaches. For our review, we decided to exhibit the key modeling hypotheses of relevant models. Formalizing these hypotheses allows us to have a clear understanding of what kind of relations cannot be modeled by a given model. Furthermore, it simplifies the usually challenging task of designing an unsupervised relation extraction loss.

As a reminder, the modeling hypotheses are listed in Appendix B.

Regularizing discriminative approaches for deep encoders.
In Chapter 3, we introduced the first unsupervised model that does not rely on hand-engineered features. In particular, we identified two critical weaknesses of previous discriminative models which hindered the use of deep neural networks. These weaknesses relate to the model's output, which tends to collapse to a trivial—either deterministic or uniform—distribution. We introduced two relation distribution losses to alleviate these problems: a skewness loss pushes the prediction away from a uniform distribution, and a distribution distance loss prevents the output from collapsing to a deterministic distribution. This allowed us to train a PCNN model to cluster unsupervised samples in clusters conveying the same relation.

Exploiting the dataset structure using graph-based models.
In Chapter 4, we investigated aggregate approaches for unsupervised relation extraction. We encoded the relation extraction problem as a graph labeling—or attributing—problem. We then showed that information can be leveraged from this structure by probing distributional regularities of random paths. To exploit this information, we designed an assumption using our experience from Chapter 2 to leverage the structure of the graph to supervise a relation extraction model. We then proposed an approach based on this hypothesis by modifying the Weisfeiler–Leman isomorphism test to use a 1-Wasserstein distance.

From a higher vantage point, we can say that we first assisted the development of deep learning approaches for the task of unsupervised relation extraction, and then helped open a new direction of research on aggregate approaches in the unsupervised setup using graph-based models. Both of these research objects were somewhat natural developments following current trends in machine learning research.

Perspectives

Using language modeling for relation extraction. A recent trend in NLP has been to encode all tasks as language models. The main embodiment of this trend is T5 (Raffel et al. 2020). T5 is trained as a masked language model (MLM, Section 1.3.4.2) on a sizeable "common crawl" of the web. Then, it is fine-tuned by prefixing the sequence with a task-specific prompt such as "translate English to German:". Relation extraction can also be trained as a text-to-text model in the supervised setup (Trisedya et al. 2019). Extending this model to the unsupervised setup—for example, through the creation of pseudo-labels—could allow us to leverage the large amount of linguistic information contained in the T5 parameters. In the same vein, Ushio et al. (2021) propose to use predefined and learned prompts for relation prediction, for example by filling in the following template: "Today, I finally discovered the relation between e_1 and e_2: e_1 is the <BLANK/> of e_2."

Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer" JMLR 2020

The name T5 comes from "Text-To-Text Transfer Transformer" since it recasts every NLP task as a text-to-text problem.

Ushio et al., "Distilling Relation Embeddings from Pretrained Language Models" 2021

More generally, relation extraction is closely related to language models. The first model we experimented on during this Ph.D. candidacy was a pre-trained language model used to fill sentences such as "The capital of Japan is <BLANK/>." While Vaswani et al. (2017) was already published at the time, pre-trained transformer language models were not widely available yet. We used a basic LSTM, which was strongly biased in favor of entities often appearing in the dataset. In practice, the model predicted "London" as the capital of most small countries. However, as we showcased in Section 2.5.6, large transformer-based models such as BERT (Devlin et al. 2019) perform well out-of-the-box on unsupervised relation extraction. An additional argument in favor of transformer-based language models comes from Chapter 3. Indeed, the *fill-in-the-blank* model seeks to predict an entity blanked in the input; this is similar to the MLM task. More abstractly, language purposes to describe a reality which can be understood—among other things—through the concept of relation. And indeed, if one understands language, one must understand the relations conveyed by language. Using a model of language as a basis for a model of relations is promising, as long as the semantic fragment of language unrelated to relations can be discarded.

Vaswani et al., "Attention is All you Need" NeurIPS 2017

Dataset-level modeling hypotheses. In the past few years, graph-based approaches have gained traction in the information extraction field (Fu et al. 2019; Qian et al. 2019) and we can only expect this interest to continue growing in the future. While knowledge of the language should be sufficient to understand the relation underlying most samples, it is challenging to design an unsupervised loss solely relying on linguistic information. Furthermore, following distributional linguistics, language—and thus relations conveyed by language—are acquired through structured repetitions. The concept of repetition captured by graph adjacency can therefore also provide a theoretical basis for the design of modeling hypotheses. We can even argue that capturing the structure of the data is an ontologically prior modeling level. For this reason, we think that relation graphs should provide a better basis for the formulation of modeling hypotheses.

Qian et al., "GraphIE: A Graph-Based Framework for Information Extraction" 2019

Complex relations. Several simplifying assumptions were made to define the relation extraction task. For example, we assume all relations to be binary, holding between exactly two entities. However, n-ary relations are needed to model complex interrelationships. For example, encoding the fact that “a drug e_1 can be used to treat a disease e_2 when the patient has genetic mutation e_3” necessitates a ternary relation. This problem has been tackled for a long time (McDonald et al. 2005; Song et al. 2018). The graph-based approaches have a natural extension to n-ary relation in the form of hypergraphs, which are graphs with n-ary edges. Since the hypergraph isomorphism problem can be polynomially reduced to the standard graph isomorphism problem (Zemlyachenko et al. 1985), we can expect n-ary extension of graph-based relation extraction approaches to work as well as standard relation extraction.

A related problem is the one of fact qualification. The fact “Versailles *capital of* France” only held until the 1789 revolution. In the Wikidata parlance, these kinds of details are called *qualifiers*. In particular, the temporal qualification can be critical to certain relation extraction datasets (Jiang et al. 2019). Some information extraction datasets already include this information (Mesquita et al. 2019); however, little work has been made in this direction yet. Qualifiers could be generated from representations of relations in a continuous manifold such as the one induced by a similarity space for few-shot evaluation. However, learning to map relation embeddings to qualifiers in an unsupervised fashion might prove difficult.

Appendix A

Résumé en français

Meta-résumé Détecter les relations exprimées dans un texte est un problème fondamental de la compréhension du langage naturel. Il constitue un pont entre deux approches historiquement distinctes de l'intelligence artificielle, celles à base de représentations symboliques et distribuées. Cependant, aborder ce problème sans supervision humaine pose plusieurs problèmes et les modèles non supervisés ont des difficultés à faire écho aux avancées des modèles supervisés. Cette thèse aborde deux lacunes des approches non supervisées : le problème de la régularisation des modèles discriminatifs et le problème d'exploitation des informations relationnelles à partir des structures des jeux de données. La première lacune découle de l'utilisation de réseaux neuronaux profonds. Ces modèles ont tendance à s'effondrer sans supervision. Pour éviter ce problème, nous introduisons deux fonctions de coût sur la distribution des relations pour contraindre le classifieur dans un état entraînable. La deuxième lacune découle du développement des approches au niveau des jeux de données. Nous montrons que les modèles non supervisés peuvent tirer parti d'informations issues de la structure des jeux de données, de manière encore plus décisive que les modèles supervisés. Nous exploitons ces structures en adaptant les méthodes non supervisées existantes pour capturer les informations topologiques à l'aide de réseaux convolutifs pour graphes. De plus, nous montrons que nous pouvons exploiter l'information mutuelle entre les données topologiques et linguistiques pour concevoir un nouveau paradigme d'entraînement pour l'extraction non supervisée de relations.

“ *Puisque tu fais de la géométrie et de la trigonométrie, je vais te donner un problème : Un navire est en mer, il est parti de Boston chargé de coton, il jauge 200 tonneaux ; il fait voile vers le Havre, le grand mât est cassé, il y a un mousse sur le gaillard d'avant, les passagers sont au nombre de douze, le vent souffle* N.-E.-E.*, l'horloge marque 3 heures un quart d'après-midi, on est au mois de mai… On demande l'âge du capitaine ?*

— Gustave Flaubert, « Lettre du 16 mai 1843 à sa sœur » (1926)

Flaubert se moque de l'enseignement mathématique à « son vieux rat » (Caroline Flaubert). Celle-ci ne répondit pas en prenant en compte la corrélation entre la responsabilité de diriger un navire jaugeant 200 tonneaux et l'avancée de la carrière du capitaine.

“ *À travers l'espace feuilleté des vingt-sept pairs, Faustroll évoqua vers la troisième dimension :*
De Baudelaire, le Silence d'Edgard Poë, en ayant soin de retraduire en grec la traduction de Baudelaire.

— Alfred Jarry, *Gestes et opinions du docteur Faustroll* (1911)

Le monde est doté d'une structure, qui nous permet de le comprendre. Cette structure est en premier lieu apparente à travers la répétition de nos expériences sensorielles. Parfois, nous voyons un chat, puis un autre chat. Les entités émergent de la répétition de l'expérience de *félinité* que nous avons ressentie. De temps en temps, nous pouvons également observer un chat *à l'intérieur* d'un carton ou une personne *à l'intérieur* d'une pièce. Les relations sont le mécanisme explicatif qui sous-tend ce deuxième type de répétition. Une relation régit une interaction entre au moins deux objets. Nous supposons qu'une relation *à l'intérieur* existe parce que nous avons vécu à plusieurs reprises la même interaction entre un conteneur et son contenu. Le vingtième siècle a été traversé par le développement du structuralisme, qui considérait que les interrelations entre phénomènes étaient plus éclairantes que l'étude des phénomènes pris isolément. En d'autres termes, nous pourrions mieux comprendre ce qu'est un chat en étudiant

Le chat du Cheshire de Tenniel (1889) vous fournit une expérience de *félinité*.

ses relations avec d'autres entités plutôt qu'en énumérant les caractéristiques de notre expérience de la *félinité*. De ce point de vue, le concept de relation est crucial dans notre compréhension du monde.

Les relations — quoique dans un sens plus restreint — sont l'un des dix *prédicaments* d'Aristote, les catégories d'objets d'appréhension humaine (GRACIA et NEWTON 2016).

Les langues naturelles saisissent la structure sous-jacente de ces répétitions à travers un processus que nous ne comprenons pas entièrement. L'un des objectifs de l'intelligence artificielle, appelé compréhension du langage naturel, est d'imiter ce processus à l'aide d'algorithmes. Puisque ce but nous échappe encore, nous nous efforçons d'en modéliser seulement des parties. Cette thèse, suivant la perspective structuraliste, se concentre sur l'extraction des relations véhiculées par la langue naturelle. En supposant que la langue naturelle est représentative de la structure sous-jacente des expériences sensorielles,[95] nous devrions être en mesure de capturer les relations en exploitant uniquement les répétitions, c'est-à-dire de manière non supervisée.

[95] Les répétitions d'expériences sensorielles et de mots n'ont pas à être nécessairement identiques. Nous ne nous préoccupons ici que de la possibilité de résoudre les références. Même si nos expériences d'arbres s'accompagnent généralement d'expériences d'écorces, les mots « arbre » et « écorce » ne cooccurrent pas aussi souvent dans des expressions en langue naturelle. Cependant, leur relation méronymique est intelligible à la fois par l'expérience d'arbres et, entre autres, par l'utilisation de la préposition « de » dans les mentions écrites d'écorces.

A.1 Contexte

L'extraction de relations peut nous aider à mieux comprendre le fonctionnement des langues. Par exemple, la question de savoir s'il est possible d'apprendre une langue à partir d'une petite quantité de données reste une question ouverte en linguistique. L'argument de la pauvreté du stimulus affirme que les enfants ne devraient pas être capable d'acquérir des compétences linguistiques en étant exposés à si peu de données.[96] Il s'agit de l'un des principaux arguments en faveur de la théorie controversée de la grammaire universelle. Capturer des relations à partir de rien d'autre qu'un petit nombre d'expressions en langue naturelle serait un premier pas vers la réfutation de l'argument de la pauvreté du stimulus.

[96] Ce qui impliquerait qu'une partie de la maîtrise du langage est innée.

Ce type de motivation derrière le problème d'extraction de relations cherche à avancer l'*épistémè*.[97] Cependant, la plupart des avancées sur cette tâche découlent d'une recherche de *technè*.[98] L'objectif final est de construire un système ayant des applications dans le monde réel. Dans cette perspective, l'intelligence artificielle a pour but de remplacer ou d'assister les humains dans des tâches spécifiques. La plupart des tâches nécessitent une certaine forme de connaissances techniques (par exemple, le diagnostic médical nécessite la connaissance des relations entre symptômes et maladies). Le principal vecteur de connaissances est le langage (par exemple, à travers l'éducation). Ainsi, l'acquisition de connaissances à partir d'énoncés en langue naturelle est un problème fondamental pour les systèmes destinés à avoir des applications concrètes.

[97] Du grec ancien ἐπιστήμη : connaissance, savoir.

[98] Du grec ancien τέχνη : technique, art.

ALEX et al. (2008) présentent une analyse de l'impact des systèmes d'extraction de connaissances à partir de textes sur un problème concret. Leur article montre que les annotateurs humains peuvent utiliser un système d'apprentissage automatique pour mieux extraire un ensemble d'interactions protéine–protéine de la littérature biomédicale. Il s'agit clairement d'une recherche de *technè* : les interactions protéine–protéine ne sont pas de nouvelles connaissances, elles sont déjà publiées ; cependant, le système améliore le travail de l'opérateur humain.

ALEX et al., "Assisted curation : does text mining really help ?" PSB 2008

Cet exemple d'application est révélateur du problème plus vaste de l'explosion informationnelle. La quantité d'informations publiées n'a cessé de croître au cours des dernières décennies. L'apprentissage automatique peut être utilisé pour filtrer ou agréger cette grande quantité de données. Pour ce genre de tâches, l'objet d'intérêt n'est pas le texte en lui-même mais la sémantique véhiculée, sa signification. Une question se pose alors : comment définir la sémantique que l'on cherche à traiter ? En effet, la définition du concept de « sens » fait l'objet de nombreuses discussions dans la communauté philosophique. Bien que certains sceptiques, comme Quine, ne reconnaissent pas le sens comme un concept essentiel, ils estiment qu'une description minimale du sens devrait au moins englober la reconnaissance de la synonymie. Cela fait suite à la discussion ci-dessus sur la reconnaissance des répétitions : si est une répétition de , nous devrions pouvoir dire que et sont synonymes. En pratique, cela implique que nous devrions être en mesure d'extraire des classes de formes linguistiques ayant la même signification ou le même référent — la différence entre les deux n'est pas pertinente pour notre problème.

Paris (Q162121) n'est ni la capitale de la France, ni le prince de Troie, c'est le genre de la parisette à quatre feuilles. La capitale de la France est Paris (Q90) et le prince de Troie, fils de Priam, Pâris (Q167646). Illustration tirée de Redouté (1802).

> *La signification, c'est ce que devient l'essence, une fois divorcée d'avec l'objet de la référence et remariée au mot.*
>
> — Willard Van Orman Quine, "Main Trends in Recent Philosophy : Two Dogmas of Empiricism" (1951)
>
> Traduction de Laugier (2004)

Bien que la discussion au sujet du sens soit essentielle pour définir la notion de relation qui nous intéresse, il est important de noter que nous travaillons sur la langue naturelle ; nous voulons extraire des relations à partir de textes, et non de répétitions d'entités abstraites. Pourtant, la correspondance entre les signifiants linguistiques et leur signification n'est pas bijective. Nous pouvons distinguer deux types de désalignement entre les deux : soit deux expressions renvoient au même objet (synonymie), soit la même expression renvoie à des objets différents selon le contexte dans lequel elle apparaît (homonymie). La première variété de désalignement est la plus courante, surtout au niveau de la phrase. Par exemple, « Paris est la capitale de la France » et « la capitale de la France est Paris » véhiculent le même sens malgré des formes écrites et orales différentes. Au contraire, le second type est principalement visible au niveau des mots. Par exemple, la préposition « de » dans les phrases « frémir de peur » et « Bellérophon de Corinthe » traduit soit une relation *causé par* soit une relation *né à*. Pour distinguer ces deux utilisations de « de, » nous pouvons utiliser des identifiants de relation tels que P828 pour *causé par* et P19 pour *né à*. Un exemple avec des identifiants d'entités — qui ont pour but d'identifier de manière unique les concepts d'entité — est donné dans la marge.

Alors que la discussion qui précède donne l'impression que tous les objets s'inscrivent parfaitement dans des concepts clairement définis, en pratique, c'est loin d'être le cas. Très tôt dans la littérature de la représentation des connaissances, Brachman (1983) a remarqué la difficulté de définir clairement des relations apparemment simples telles que *instance de* (P31). Ce problème découle de l'hypothèse selon laquelle la synonymie est transitive et, par conséquent, induit des classes d'équivalence. Cette hypothèse est assez naturelle puisqu'elle s'applique déjà au lien entre le langage et ses références : même si deux chats peuvent être très différents l'un de l'autre, nous les regroupons sous le même signifiant. Cependant, la langue naturelle est flexible. Lorsque nous essayons de capturer l'entité « chat, » il n'est pas tout à fait clair si nous incluons « un chat avec le corps d'une tarte aux cerises » dans les expériences ordinaires de chat.[99] Pour contourner ce problème, certains travaux récents sur le problème d'extraction de relations (Han et al. 2018) définissent la synonymie comme une association continue intransitive. Au lieu de regrouper les formes linguistiques dans des classes bien définies partageant un sens unique, ils extraient une fonction de similarité mesurant la ressemblance de deux objets.

Brachman, "What is-a Is and Isn't : An Analysis of Taxonomic Links in Semantic Networks" Computer 1983

[99] Le lecteur qui décrirait une telle entité comme étant un chat est invité à remplacer diverses parties du corps de ce chat imaginaire par des aliments jusqu'à ce que cesse son expérience de *félinité.*

Han et al., "FewRel : A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation" emnlp 2018

Maintenant que nous avons conceptualisé notre problème, concentrons-nous sur l'approche technique que nous proposons. Tout d'abord, pour résumer, cette thèse se concentre sur l'extraction non supervisée de relations à partir de textes.[100] Les relations étant des objets capturant les

[100] Nous utilisons le texte car il s'agit de l'expression la moins ambiguë et la plus facile à traiter de la langue.

interactions entre les entités, notre tâche est de trouver la relation reliant deux entités données dans un texte. Par exemple, dans les trois exemples suivants où les entités sont soulignées :

> Megrez$_{e_1}$ est une étoile de la constellation circumpolaire nord de la Grande Ourse$_{e_2}$.
>
> Posidonios$_{e_1}$ était un philosophe, astronome, historien, mathématicien et professeur grec originaire d'Apamée$_{e_2}$.
>
> Hipparque$_{e_1}$ est né à Nicée$_{e_2}$, et est probablement mort sur l'île de Rhodes, en Grèce.

nous souhaitons reconnaître que les deux dernières phrases véhiculent la même relation — dans ce cas, e_1 *né à* e_2 (P19) — ou du moins, suivant la discussion du paragraphe précédent sur la difficulté de définir des classes de relations, nous voulons reconnaître que les relations exprimées par les deux derniers échantillons sont plus proches l'une de l'autre que celle exprimée par le premier échantillon. Nous avançons que cela peut être réalisé par des algorithmes d'apprentissage automatique. En particulier, nous étudions comment aborder cette tâche en utilisant l'apprentissage profond. Bien que l'extraction de relations puisse être abordée comme un problème de classification supervisée standard, l'étiquetage d'un jeu de données avec des relations précises est une tâche fastidieuse, en particulier lorsque l'on traite des documents techniques tels que la littérature biomédicale étudiée par Alex et al. (2008). Un autre problème fréquemment rencontré par les annotateurs est la question de l'applicabilité d'une relation, par exemple, l'expression « le père$_{e_2}$ fondateur du pays$_{e_1}$ » doit-elle être étiquetée avec la relation *produit–producteur* ?[101] Nous examinons maintenant comment l'apprentissage profond est devenu la technique la plus prometteuse pour s'attaquer aux problèmes de traitement de la langue naturelle.

La matière première du problème d'extraction de relations est le langage. Le traitement automatique de la langue naturelle (TAL)[102] était déjà une direction de recherche importante dans les premières années de l'intelligence artificielle. On peut le voir du point de vue *épistémè* dans l'article fondateur de Turing (1950). Cet article propose la maîtrise du langage comme preuve d'intelligence, dans ce qui est maintenant connu sous le nom de test de Turing. La langue était également un sujet d'intérêt pour des objectifs de *technè*. En janvier 1954, l'expérience de Georgetown–IBM tente de démontrer la possibilité de traduire le russe en anglais à l'aide d'ordinateurs (Dostert 1955). L'expérience proposait de traduire soixante phrases en utilisant un dictionnaire bilingue pour traduire individuellement les mots et six types de règles grammaticales pour les réorganiser. Les premières expériences ont suscité beaucoup d'attentes, qui ont été suivies d'une inévitable déception, entraînant un « hiver » durant lequel les fonds attribués à la recherche en intelligence artificielle ont été restreints. Si la traduction mot à mot est assez facile dans la plupart des cas, la traduction de phrases entières est beaucoup plus difficile. La mise à l'échelle de l'ensemble des règles grammaticales dans l'expérience de Georgetown–IBM s'est avérée impraticable. Cette limitation n'était pas d'ordre technique. Avec l'amélioration des systèmes de calcul, davantage de règles auraient pu facilement être codées. L'un des problèmes identifiés à l'époque était celui de la compréhension du sens commun.[103] Pour traduire ou, plus généralement, traiter une phrase, il faut la comprendre dans le contexte du monde dans lequel elle a été prononcée. De simples règles de réécriture ne peuvent pas rendre compte de ce processus.[104] Pour pouvoir traiter des

Nous utilisons les identifiants Wikidata (`https://www.wikidata.org`) pour indexer les entités et les relations. Les identifiants des entités commencent par `Q`, tandis que les identifiants des relations commencent par `P`. Par exemple, Q35120 est une entité.

Ariane se réveille sur le rivage de Naxos où elle a été abandonnée, peinture murale d'Herculanum dans la collection du British Museum (100 av. n. è.-100 de n. è.). Le navire au loin peut être identifié comme étant le bateau de Thésée, pour l'instant. Selon le point de vue philosophique du lecteur (Q1050837), son identité en tant que bateau de Thésée pourrait ne pas perdurer.

[101] L'annotateur de ce morceau de phrase dans le jeu de données SemEval 2010 Task 8 a considéré qu'il exprimait effectivement la relation *produit–producteur*. La difficulté d'appliquer précisément une définition est un argument supplémentaire en faveur des approches basées sur les fonctions de similarité par rapport aux approches de classification.

[102] *natural language processing* (NLP)

Turing, "Computing Machinery and Intelligence" Mind 1950

[103] *commonsense knowledge*

[104] Par ailleurs, la grammaire est encore un domaine de recherche actif. Nous ne comprenons pas parfaitement la réalité sous-jacente capturée par la plupart des mots et sommes donc incapables d'écrire des règles formelles complètes pour leurs usages. Par exemple, Marque-Pucheu (2008) présente un article de linguistique traitant de l'utilisation des prépositions françaises « de » et « à. » C'est l'un des arguments en faveur des approches non supervisées ; en évitant d'étiqueter manuellement les jeux de données, nous évitons la limite des connaissances des annotateurs humains.

phrases entières, un changement de paradigme était nécessaire.

Une première évolution a eu lieu dans les années 1990 avec l'avènement des approches statistiques (S. Abney 1996). Ce changement peut être attribué en partie à l'augmentation de la puissance de calcul, mais aussi à l'abandon progressif de préceptes linguistique essentialistes au profit de préceptes distributionnalistes.[105] Au lieu de s'appuyer sur des experts humains pour concevoir un ensemble de règles, les approches statistiques exploitent les répétitions dans de grands corpus de textes pour déduire ces règles automatiquement. Par conséquent, cette progression peut également être considérée comme une transformation des modèles d'intelligence artificielle symbolique vers des modèles statistiques. La tâche d'extraction de relations a été formalisée à cette époque. Et si les premières approches étaient basées sur des modèles symboliques utilisant des règles prédéfinies, les méthodes statistiques sont rapidement devenues la norme après les années 1990. Cependant, ces modèles statistiques reposaient toujours sur des connaissances linguistiques. Les systèmes d'extraction de relations étaient généralement divisés en une première phase d'extraction de caractéristiques linguistiques spécifiées à la main et une seconde phase où une relation était prédite à partir de ces caractéristiques à l'aide de modèles statistiques peu profonds.

[105] Noam Chomsky, l'un des linguistes essentialistes les plus importants, considère que la manipulation de probabilités d'extraits de texte ne permet pas d'acquérir une meilleure compréhension du langage. Suite au succès des approches statistiques, il n'a reconnu qu'un accomplissement de *technè* et non d'*épistémè*. Pour une réponse à cette position, voir S. Abney (1996) et Norvig (2011).

> *Cheval blanc n'est pas cheval.* 「白馬非馬」
> — "Gongsun Longzi" Chapitre 2 (circa 300 av. n. è.)

Un paradoxe bien connu de la philosophie chinoise illustrant la difficulté de définir clairement le sens véhiculé par la langue naturelle. Ce paradoxe peut être résolu en désambiguïsant le mot « cheval. » Fait-il référence à « l'ensemble de tous les chevaux » (la vision méréologique) ou à « la chevalité » (la vision platonicienne) ? L'interprétation méréologique a été célèbrement — et de manière controversée — introduite par Hansen (1983), voir Fraser (2007) pour une discussion des premières vues ontologiques du langage en Chine.

Frontispice de la bibliothèque OuCuiPienne par Chevalier (1990). Une autre façon de cuisiner avec les lettres.

Une deuxième évolution est survenue dans les années 2010 lorsque les approches d'apprentissage profond ont effacé la séparation entre les phases d'extraction de caractéristiques et de prédiction. Les modèles d'apprentissage profond sont entrainés pour traiter directement les données brutes, dans notre cas des extraits de texte. À cette fin, des réseaux de neurones capables d'approcher n'importe quelle fonction sont utilisés. Cependant, l'entraînement de ces modèles nécessite généralement de grandes quantités de données étiquetées. Il s'agit d'un problème particulièrement important pour nous puisque nous traitons un problème non supervisé. En tant que technique la plus récente et la plus efficace, l'apprentissage profond est un choix naturel pour s'attaquer à l'extraction de relations. Cependant, ce choix s'accompagne de problématiques que nous essayons de résoudre dans ce manuscrit.

A.2 Régularisation des modèles discriminatifs d'extraction non supervisée de relations

L'évolution des méthodes d'extraction de relations non supervisées suit de près celle des méthodes de tal décrite ci-dessus. La première approche utilisant des techniques d'apprentissage profond a été celle de Marcheggiani et Titov (2016). Cependant, une partie de leur modèle reposait toujours sur des caractéristiques linguistiques extraites en amont. La raison pour laquelle cette extraction ne pouvait pas être faite automatiquement, comme c'est habituellement le cas en apprentissage profond, est étroitement liée à la nature non supervisée du problème. Notre première contribution est de proposer une technique permettant l'entraînement d'approches d'extraction non supervisée de relations par apprentissage profond.

Nous avons identifié deux problèmes critiques des modèles discriminants existant qui entravent l'utilisation de réseaux neuronaux profonds pour l'extraction de caractéristiques. Ces problèmes concernent la sortie du classifieur, qui a tendance à s'effondrer en une distribution triviale, soit déterministe, soit uniforme. Nous proposons d'introduire deux fonctions de coût sur la distribution des relations pour atténuer ces problèmes : une fonction d'asymétrie éloigne la prédiction d'une loi uniforme, et une distance de distributions empêche la sortie de s'effondrer vers une distribution déterministe. Cela nous a permis d'entraîner un modèle PCNN (Zeng et al. 2015) pour regrouper les échantillons non supervisés en partitions[106] véhiculant la même relation.

Notre approche se base sur le problème de remplissage de texte à trous :

"Le $\underline{\text{sol}}_{e_1}$ a été la monnaie du $\underline{\ ?\ }_{e_2}$ entre 1863 et 1985."

Pour pouvoir remplir cette phrase avec le mot manquant, il est nécessaire de comprendre la relation véhiculée. Nous utilisons cette tâche comme un substitut nous permettant d'identifier la sémantique relationnelle de la phrase. Étant donné une phrase s contenant deux entités $\boldsymbol{e}$ exprimant la relation r, nous modélisons la probabilité suivante :

$$P(e_{-i} \mid s, e_i) = \sum_{r\in\mathcal{R}} \underbrace{P(r \mid s)}_{\text{(i) classifieur}} \underbrace{P(e_{-i} \mid r, e_i)}_{\text{(ii) prédicteur d'entité}} \qquad \text{pour } i = 1, 2.$$

Nous utilisons un réseau profond (PCNN, Zeng et al. 2015) pour le classifieur et le même modèle que Marcheggiani et Titov (2016) pour la prédiction d'entité. Le modèle résultant présente des instabilités, comme celle illustrée par la Figure A.1. Nous proposons deux fonctions de coût supplémentaires sur les paramètres $\boldsymbol{\phi}$ du classifieur pour résoudre ces problèmes :

$$\mathcal{L}_{\text{S}}(\boldsymbol{\phi}) = \mathop{\mathbb{E}}_{(s,\boldsymbol{e})\sim\mathcal{U}(\mathcal{D})} [\mathrm{H}(\mathrm{R} \mid s, \boldsymbol{e}; \boldsymbol{\phi})]$$
$$\mathcal{L}_{\text{D}}(\boldsymbol{\phi}) = \mathrm{D}_{\text{KL}}(P(\mathrm{R} \mid \boldsymbol{\phi}) \parallel \mathcal{U}(\mathcal{R})).$$

La première fonction force la sortie du classifieur a avoir une entropie faible ce qui résout le problème de la Figure A.1. La seconde fonction s'assure qu'une variété de relations soient prédites pour différents échantillons. Ces deux fonctions nous permettent d'entrainer un réseau profond pour l'extraction non supervisée de relations comme le montrent les scores de la Table A.1.

A.3 Modélisation à l'aide de graphes de la structure des jeux de données

Comme mentionné dans la Section A.1, les approches récentes utilisent une définition plus souple des relations en extrayant une fonction de similarité au lieu d'un classifieur. De plus, elles considèrent un contexte plus large : au lieu de traiter chaque phrase individuellement, la cohérence globale des relations extraites est prise en compte. Cependant, ce deuxième type d'approches a principalement été appliqué au cadre supervisé, avec une utilisation plus limitée dans le cadre non supervisé. Notre deuxième contribution concerne l'utilisation de ce contexte plus large pour l'extraction non supervisée de relations. En particulier, nous établissons des parallèles avec le test d'isomorphisme de Weisfeiler–Leman pour concevoir de nouvelles méthodes utilisant conjointement des caractéristiques topologiques (au niveau des jeux de données) et linguistiques (au niveau des phrases).

Cette section a fait l'objet d'une publication :
Étienne Simon, Vincent Guigue, Benjamin Piwowarski. "Unsupervised Information Extraction : Regularizing Discriminative Approaches with Relation Distribution Losses" ACL 2019

Zeng et al., "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks" EMNLP 2015

[106] *clusters*

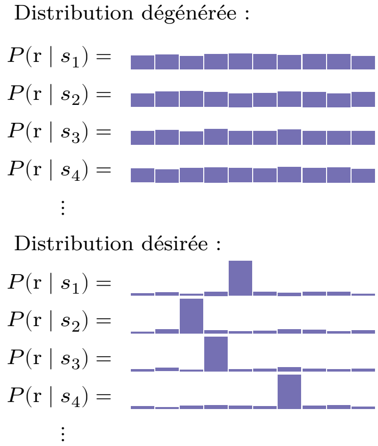


Figure A.1 : Illustration du problème d'uniformité. Le classifieur attribue la même probabilité à toutes les relations. À la place, nous souhaitons que le classifieur prédise clairement une relation unique pour chaque échantillon.

Modèle		$\mathrm{B}^3 F_1$
Classif.	Reg.	
Linear	$\mathcal{L}_{\text{VAE REG}}$	35,2
PCNN	$\mathcal{L}_{\text{VAE REG}}$	27,6
Linear	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	37,5
PCNN	$\mathcal{L}_{\text{S}} + \mathcal{L}_{\text{D}}$	**39,4**

Table A.1 : Résultats quantitatifs des méthodes de partitionnement sur le jeu de données NYT-FB. On distingue le classifieur utilisé (Classif.) de la régularisation utilisée (Reg.). La régularisation $\mathcal{L}_{\text{VAE REG}}$ est celle issue de l'article de Marcheggiani et Titov (2016).

Nous encodons le problème d'extraction de relations comme un problème d'étiquetage d'un multigraphe $G = (\mathcal{E}, \mathcal{A}, \boldsymbol{\varepsilon}, \rho, \varsigma)$ défini comme suit :

- $\mathcal{E}$ est l'ensemble des nœuds qui correspondent aux entités.
- $\mathcal{A}$ est l'ensemble des arcs qui connectent deux entités.
- $\varepsilon_1 : \mathcal{A} \to \mathcal{E}$ associe à chaque arc son nœud d'origine (l'entité marquée e_1),
- $\varepsilon_2 : \mathcal{A} \to \mathcal{E}$ associe à chaque arc son nœud de destination (l'entité marquée e_2),
- $\varsigma : \mathcal{A} \to \mathcal{S}$ associe à chaque arc $a \in \mathcal{A}$ la phrase correspondante contenant $\varepsilon_1(a)$ et $\varepsilon_2(a)$,
- $\rho : \mathcal{A} \to \mathcal{R}$ associe à chaque arc $a \in \mathcal{A}$ la relation entre les deux entités véhiculée par $\varsigma(a)$.

Étant donné un chemin dans ce graphe :

$$\mathrm{e}_1 \xrightarrow{\mathrm{r}_1} \mathrm{e}_2 \xrightarrow{\mathrm{r}_2} \mathrm{e}_3 \xrightarrow{\mathrm{r}_3} \mathrm{e}_4,$$

nous avons conçu un algorithme de comptage basé sur l'exponentiation de la matrice d'adjacence de G et sur un échantillonnage préférentiel[107] qui nous permet d'approcher l'information mutuelle $\mathrm{I}(\mathrm{r}_2; \mathrm{r}_1, \mathrm{r}_3) \approx 6{,}95$ bits. Elle se décompose en une entropie conditionnelle $\mathrm{H}(\mathrm{r}_2 \mid \mathrm{r}_1, \mathrm{r}_3) \approx 1{,}06$ bits soustrait à l'entropie croisée[108] $\mathbb{E}_{r_1,r_3}[\mathrm{H}_{P(\mathrm{r}_2)}(\mathrm{r}_2 \mid r_1, r_3)] \approx 8{,}01$ bits. Cela signifie que la majeure partie de l'information relationnelle est extractible à partir du voisinage dans le graphe G.

Fort de cette observation, nous utilisons l'hypothèse suivante pour concevoir un nouveau paradigme pour l'extraction non supervisée de relations :

Hypothèse distributionnelle faible sur le graphe d'extraction de relations. *Deux arcs véhiculent des relations similaires s'ils ont des voisinages similaires.*

Pour exploiter cette information de voisinage présente dans la topologie du multigraphe G, nous proposons de nous inspirer du test d'isomorphisme de Weisfeiler–Leman (WL, WEISFEILER et LEMAN 1968). Deux graphes sont dits isomorphes s'il existe un morphisme entre leur sommets qui conserve la relation de voisinage. Ce concept est illustré par la Figure A.2. Nous pouvons donc traduire l'hypothèse ci-dessus par l'affirmation que si les voisinages de deux échantillons sont isomorphes, alors ces deux échantillons véhiculent la même relation. Pour évaluer la proximité de deux voisinages, nous définissons $\mathfrak{S}_{\rightarrow}(a, k)$, le plongement par BERTcoder (voir Figure A.3) de la sphère de rayon k autour de l'arête $a \in \mathcal{A}$ comme :

$$\begin{aligned}
S_{\rightarrow}(a, 0) &= \{\, a \,\} \\
S_{\rightarrow}(a, k) &= \{\, x \in \mathcal{A} \mid \exists y \in S_{\rightarrow}(a, k-1) : \varepsilon_1(x) = \varepsilon_2(y) \,\} \\
\mathfrak{S}_{\rightarrow}(a, k) &= \{\, \text{BERTcoder}(\varsigma(x)) \in \mathbb{R}^d \mid x \in S_{\rightarrow}(a, k) \,\}.
\end{aligned}$$

Ces sphères correspondent au voisinage à distance k. À partir de celles-ci, nous pouvons définir une fonction de distance prenant en compte le voisinage jusqu'à une distance K :

$$d(a, a'; \boldsymbol{\lambda}) = \sum_{k=0}^{K} \frac{\lambda_k}{2} \sum_{o \in \{\leftarrow, \rightarrow\}} W_1\left(\mathfrak{S}_o(a, k), \mathfrak{S}_o(a', k)\right),$$

où W_1 désigne la distance de Wasserstein d'ordre 1. En particulier, cette fonction évaluée en $\boldsymbol{\lambda} = [1]$ correspond à la distance habituelle entre plongements de phrases modulo l'utilisation de W_1 à la place d'une distance

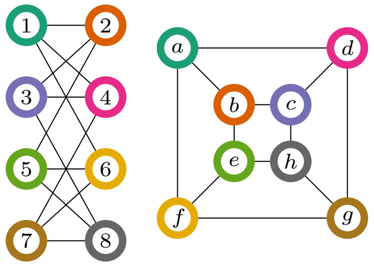


FIGURE A.2 : Exemple de graphes isomorphes. Chaque nœud i dans le graphe de gauche correspond à la i-ième lettre de l'alphabet dans le graphe de gauche. Par ailleurs, ces graphes contiennent des automorphismes non-triviaux, par exemple en associant le nœud i au nœud $9 - i$.

[107] *importance sampling*

[108] *cross-entropy*

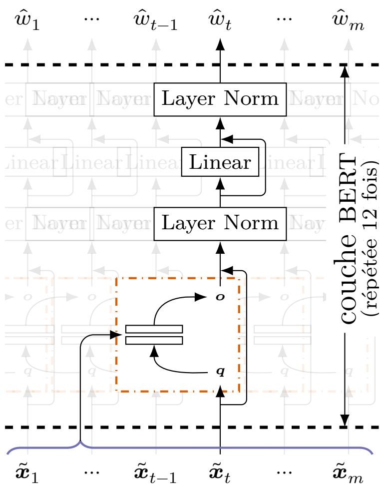


FIGURE A.3 : Schéma de BERT (DEVLIN et al. 2019), un modèle de langue masqué basé sur un *transformer*. Le modèle est entraîné à reconstruire des mots $\hat{w}_t$ corrompus en $\tilde{w}_t$ (plongés en $\tilde{\boldsymbol{x}}_t$). BERTcoder est une spécialisation de ce modèle pour l'extraction de relations (SOARES et al. 2019).

KIPF et WELLING (2017) ont déjà tracé un parallèle entre WL et les approches à base de réseaux neuronaux convolutifs pour graphes (GCN). Toutefois, nous avançons que les fonctions d'apprentissage habituellement utilisées pour les GCN ne sont pas adaptées au problème d'extraction non supervisée de relations.

cosinus. Pour des raisons de limites de calcul, nous fixons $K = 2$. Dans ce cas, $d(a_1, a_2, [1, 0]^\intercal)$ correspond à la distance linguistique entre deux échantillons $a_1, a_2 \in \mathcal{A}$, tandis que $d(a_1, a_2, [0, 1]^\intercal)$ correspond à la distance topologique entre les voisinages des échantillons a_1 et a_2. Nous proposons de faire coïncider ces deux distances pour tirer parti de l'information mutuelle au voisinage et à la phrase afin d'identifier la sémantique relationnelle des échantillons. Pour ce faire, nous introduisons une fonction de coût par triplet :[109]

$$\mathcal{L}_{\text{LT}}(a_1, a_2, a_3) = \max\begin{pmatrix} 0, \zeta + 2\big(d(a_1, a_2, [1,0]^\intercal) - d(a_1, a_2, [0,1]^\intercal)\big)^2 \\ \qquad - \big(d(a_1, a_2, [1,0]^\intercal) - d(a_1, a_3, [0,1]^\intercal)\big)^2 \\ \qquad - \big(d(a_1, a_3, [1,0]^\intercal) - d(a_1, a_2, [0,1]^\intercal)\big)^2 \end{pmatrix}.$$

[109] *triplet loss*

Modèle	Précision
Linguistique (BERT)	69,46
Topologique (W_1)	65,75
Tous les deux	72,18

Table A.2 : Résultats quantitatifs des méthodes à base de graphe sur le jeu de données FewRel (Han et al. 2018). Ces résultats portent uniquement sur les échantillons de FewRel connectés par au moins une arête dans le graphe G du jeu de données T-REx.

Des résultats préliminaires sur l'utilisation d'informations topologiques sont donnés dans la Table A.2. Comme on pouvait s'y attendre, l'information relationnelle encodée dans le voisinage d'ordre 1 du graphe est moindre que celle directement contenue dans la phrase. Toutefois, ces informations peuvent être combinées ce qui permet d'améliorer significativement la performance du modèle d'extraction de relation.

A.4 Conclusion

Pendant ma candidature au doctorat, je me suis—principalement[110]—concentré sur l'étude de l'extraction non supervisée de relations. Dans cette tâche, étant donné un ensemble de phrases et de paires d'entités, nous recherchons l'ensemble des faits véhiculés (e_1, r, e_2), tels que r exprime la relation entre e_1 et e_2 dans un échantillon. Pour mener à bien cette tâche, nous avons suivi deux axes de recherche principaux : premièrement, la question de savoir comment entraîner un réseau neuronal profond pour l'extraction non supervisée de relations ; deuxièmement, la question de savoir comment tirer parti de la structure d'un ensemble de données pour obtenir des informations supplémentaires pour la tâche d'extraction de relations sans supervision.

[110] Avec la distraction occasionnelle—et profondément appréciée—de Syrielle Montariol sur d'autres projets de TAL (Montariol et al. 2022).

Plus grossièrement, nous avons d'abord aidé au développement d'approches d'apprentissage profond pour la tâche d'extraction non supervisée de relations, puis contribué à ouvrir une nouvelle direction de recherche sur les approches au niveau des jeux de données dans la configuration non supervisée utilisant des modèles basés sur des graphes. Ces deux objets de recherche étaient en quelque sorte des développements naturels suivant les tendances actuelles de la recherche en apprentissage automatique.

Appendix B

List of Assumptions

Modeling hypotheses are central to relation extraction approaches, especially unsupervised ones (see Chapter 2). This appendix list all assumptions introduced in the previous chapters in alphabetical order, with reference to the section in which it was introduced, and whenever possible a counterexample exposing what kind of construct cannot be captured by making this hypothesis.

Assumption $\mathscr{H}_{1\rightarrow 1}$: *All relations are one-to-one.* $1 \rightarrow 1$

$\forall r \in \mathcal{R}\colon r \bullet \breve{r} \cup \boldsymbol{I} = \breve{r} \bullet r \cup \boldsymbol{I} = \boldsymbol{I}$

Appeared Section 2.5.6.
Counterexample: "Josetsu *born in* Kyushu" and "Minamoto no Shunrai *born in* Kyushu."

Assumption $\mathscr{H}_{\text{1-ADJACENCY}}$: *There is no more than one relation linking any two entities.* 1-ADJACENCY

$\forall r_1, r_2 \in \mathcal{R}\colon r_1 \cap r_2 = \mathbf{0}$

Appeared Section 2.3.2.
Counterexample: "Khayyam *born in* Nishapur" and "Khayyam *died in* Nishapur."

Assumption $\mathscr{H}_{\text{1-NEIGHBORHOOD}}$: *Two samples with the same neighborhood in the relation extraction graph convey the same relation.* 1-NEIGHBORHOOD

$\forall a, a' \in \mathcal{A}\colon \mathcal{N}(a) = \mathcal{N}(a') \implies \rho(a) = \rho(a')$

Appeared Section 4.4.3.
Counterexample: *born in* and *died in*. Since the arc-neighborhood $\mathcal{N}$ is split between in-and out-neighborhood, this hypothesis is close to $\mathscr{H}_{\text{TYPE}}$. The main difference being that the partitions (types) of $\mathscr{H}_{\text{TYPE}}$ can't overlap. While a relation which can have any type as a subject can't be modeled under the $\mathscr{H}_{\text{TYPE}}$ hypothesis, it will simply correspond to a distribution with mass on all entities in the $\mathscr{H}_{\text{1-NEIGHBORHOOD}}$ assumption.

Assumption $\mathscr{H}_{\text{BICLIQUE}}$: *Given a relation, the entities are independent of one another:* $\mathrm{e}_1 \perp\!\!\!\perp \mathrm{e}_2 \mid \mathrm{r}$. *In other words, given a relation, all possible head entities are connected to all possible tail entities.* BICLIQUE

$\forall r \in \mathcal{R} : \exists A, B \subseteq \mathcal{E} : r \bullet \breve{r} = \mathbf{1}_A \wedge \breve{r} \bullet r = \mathbf{1}_B$

Appeared Section 2.5.4.
Counterexample: most relations should infringe this assumption since it is decomposable into two unary predicates: whether the entity is part of A and whether it is part of B. For example "Alonzo Church *died in* Hudson" and "Alan Turing *died in* Wilmslow" are true but "Alonzo Church *died in* Wilmslow" is false.

Assumption $\mathscr{H}_{\text{BLANKABLE}}$: *The relation can be predicted by the text surrounding the two entities alone. Formally, using* blanked(s) *to designate the tagged sentence $s \in \mathcal{S}$ from which the entities surface forms were removed, we can write:* BLANKABLE

$\mathrm{r} \perp\!\!\!\perp \mathbf{e} \mid \text{blanked(s)}.$

Appeared Section 3.1.0.
Counterexample: some surface forms are mapped to different relations depending on the nature of the entities; in FewRel, "$\underline{\ ?\ }_{e_1}$ is part of $\underline{\ ?\ }_{e_2}$" can both convey *part of* and *part of constellation*.

Assumption $\mathscr{H}_{\text{CTX(1-ADJACENCY)}}$: *Two samples with the same contextualized representation of their entities' surface forms convey the same relation.* CTX(1-ADJACENCY)

$$\forall (s, \boldsymbol{e}, r), (s', \boldsymbol{e}', r') \in \mathcal{D}_{\mathcal{R}}: \quad \operatorname{ctx}_1(s) = \operatorname{ctx}_1(s') \wedge \operatorname{ctx}_2(s) = \operatorname{ctx}_2(s') \implies r = r'$$

Appeared Section 2.5.7.
Finding a counterexample for this assumption is quite difficult since it depends on the operation performed by the contextualization function ctx. In this sense, it is a weak assumption.

Assumption $\mathscr{H}_{\text{DISTANT}}$: *A sentence conveys all the possible relations between all the entities it contains.* DISTANT

$\mathcal{D}_{\mathcal{R}} = \mathcal{D} \bowtie \mathcal{D}_{\text{KB}}$

where $\bowtie$ denotes the natural join operator:

$$\mathcal{D} \bowtie \mathcal{D}_{\text{KB}} = \{\, (s, e_1, e_2, r) \mid (s, e_1, e_2) \in \mathcal{D} \wedge (e_1, e_2, r) \in \mathcal{D}_{\text{KB}} \,\}.$$

Appeared Section 2.2.2.
Counterexample: "Chekhov found himself coughing blood, and in 1886 the attacks worsened, but he would not admit his tuberculosis to his family or his friends." does not convey the fact "Anton Chekhov *cause of death* Tuberculosis," it only conveys "Anton Chekhov *has medical condition* Tuberculosis."

Assumption $\mathscr{H}_{\text{MULTI-INSTANCE}}$: *All facts $(\boldsymbol{e}, r) \in \mathcal{D}_{\text{KB}}$ are conveyed by at least one sentence of the unlabeled dataset $\mathcal{D}$.* MULTI-INSTANCE

$\forall (e_1, e_2, r) \in \mathcal{D}_{\text{KB}} : \exists (s, e_1, e_2) \in \mathcal{D} : (s, e_1, e_2) \text{ conveys } e_1\ r\ e_2$

Appeared Section 2.4.2.
Counterexample: Even though "Josetsu *born in* Kyushu" is present in Wikidata, at the time of writing, this information is missing from its English Wikipedia page, thus an alignment of $\mathcal{D} =$ Wikipedia with $\mathcal{D}_{\text{KB}} =$ Wikidata would not verify $\mathscr{H}_{\text{MULTI-INSTANCE}}$.

Assumption $\mathscr{H}_{\text{PULLBACK}}$: *It is possible to find the relation conveyed by a sample by looking at the entities alone and ignoring the sentence; and conversely by looking at the sentence alone and ignoring the entities.* PULLBACK

$\mathcal{D} = \mathcal{S} \times_{\mathcal{R}} \mathcal{E}^2$.

Appeared Section 2.2.1.
Entails $\mathscr{H}_{\text{1-ADJACENCY}}$.
Counterexample: Unless the reader is familiar with biographies of early Chinese philosophers, the relation between Q1362266 "Gongsun Long" and Q197430 "Zhao" should not be immediately obvious.

Assumption $\mathscr{H}_{\text{TYPE}}$: *All entities have a unique type, and all relations are left and right restricted to one of these types.* TYPE

$\exists \mathcal{T}$ partition of $\mathcal{E} : \forall r \in \mathcal{R} : \exists X, Y \in \mathcal{T} : r \bullet \breve{r} \cup \mathbf{1}_X = \mathbf{1}_X \wedge \breve{r} \bullet r \cup \mathbf{1}_Y = \mathbf{1}_Y$

Appeared Section 2.5.3.
Counterexample: "Deneb *part of* Summer Triangle" (type pair: star–constellation) and "Mitochondrion *part of* Cytoplasm" (type pair: organelle–cellular component).

Assumption $\mathscr{H}_{\text{UNIFORM}}$: *All relations occur with equal frequency.* UNIFORM

$\forall r \in \mathcal{R}\colon P(r) = \dfrac{1}{|\mathcal{R}|}$

Appeared Section 2.5.5.
Counterexample: The relation "*worshipped by*" generally appears quite a lot less than "*place of burial*" whether measured through the number of facts in Wikidata or as the number of sentences conveying these relations in Wikipedia.

Appendix C

Datasets

In this appendix, we present the primary datasets used throughout this thesis. Each section corresponds to a dataset or group of datasets. We focus on the peculiarities which make each dataset unique and provide some statistics relevant to our task.

C.1 ACE

Automatic content extraction (ACE) is a NIST program that developed several datasets for the evaluation of entity chunking and relation extraction. It is the spiritual successor of MUC (Section C.4). In their nomenclature, the task of relation extraction is called relation detection and categorization (RDC). Datasets for relation extraction were released yearly between 2002 and 2005.[111] This makes comparison difficult; for example, in Chapter 2, we mention an ACE dataset for several models (Sections 2.3.4, 2.3.5, 2.4.1 and 2.4.5); however, the versions of the datasets differs.

[111] The dataset from September 2002 is called ACE-2. This refers to the "second phase" of ACE. The pilot and first phase corpora only dealt with entity detection.

A peculiarity of the ACE dataset is its hierarchy of relations. For example, the ACE-2003 dataset contains a *social* relation type, which is divided into several relation subtypes such as *grandparent* and *sibling*. Results can be reported either on the relation types or subtypes, usually using an F_1 measure or a custom metric designed by ACE (Doddington et al. 2004) to handle directionality and the "*other*" relation (Section 2.1.1.1).

Doddington et al., "The automatic content extraction (ACE) program-tasks, data, and evaluation." LREC 2004

C.2 FewRel

FewRel (Han et al. 2018) is a few-shot relation extraction dataset. Given a query and several candidates, the model must decide which candidate conveys the relation closest to the one conveyed by the query. Therefore, FewRel is used to evaluate continuous relation representations; it is not typically used to evaluate a clustering model. For details on the few-shot setup, refer to Section 2.5.1.2.

Han et al., "FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation" EMNLP 2018

The dataset was first constructed by aligning Wikipedia with Wikidata (Section C.8) using distant supervision (Section 2.2.2). Human annotators then hand-labeled the samples. The resulting dataset is perfectly balanced; all relations are represented by precisely 700 samples. The set of the 100 most common relations with good inter-annotator agreement was then divided into three splits, whose sizes are given in Table C.1. Since common relations were strongly undersampled to obtain a balanced dataset, entities do not repeat much. The attributed multigraph (Section 4.1) corresponding to the train split of FewRel is composed of several connected components. The larger one covers approximately 21% of the vertices, while more than half of all vertices are in connected components of size three or less.

Split	Relations	Samples
Train	64	44 800
Valid	16	11 200
Test	20	14 000

Table C.1: Statistics of the FewRel dataset. The test relations and samples are not publicly available.

FewRel can be used for n way k shot evaluation, where usually $n \in \{5, 10\}$ and $k \in \{1, 5\}$. For reference, Han et al. (2018) provides human performance on 5 way 1 shot (92.22% accuracy) and 10 way 1 shot (85.88% accuracy).

A subsequent dataset released by the same team called FewRel 2.0 (Gao et al. 2019) revisited the task by adding two variations:

Gao et al., "FewRel 2.0: Towards More Challenging Few-Shot Relation Classification" EMNLP 2019

Domain adaptation, the training set of the original FewRel is used (Wikipedia–Wikidata), but the model is evaluated on biomedical literature (PubMed–UMLS) containing relations such as *may treat* and *manifestation of.*

Detecting *other* relation, also called none-of-the-above, when the relation conveyed by the query does not appear in the candidates.

While domain adaptation is an interesting problem, for unsupervised approaches, the detection of *other* seems to defeat the point of modeling a similarity space instead of clustering relations. Furthermore, we only use FewRel as an evaluation tool and never train on it; using this second dataset made, therefore, little sense.

C.3 Freebase

Freebase (Bollacker et al. 2008) is a knowledge base (Section 1.4) started in 2007 and discontinued in 2016. As one of the first widely available knowledge bases containing general knowledge, Freebase was widely used for weak supervision. In particular, it is the knowledge base used in the original distant supervision article (Mintz et al. 2009). Freebase was a collaborative knowledge base; as such, its content evolved through its existence. Therefore, even though Mintz et al. (2009), Yao et al. (2011) and Marcheggiani and Titov (2016) all run experiments on Freebase, their results are not comparable since they use different versions of the dataset. Data dumps are still provided by Google (2016); however, most of the facts were transferred to the Wikidata knowledge base (Section C.8). Some statistics about the latest version of Freebase are provided in Table C.2. However, note that most relations in Freebase are scarcely used; only 6 760 relations appear in more than 100 facts. Furthermore, the concept of entities is quite wide in Freebase, in particular it makes use of a concept called mediator (Chah 2017):

Bollacker et al., "Freebase: a collaboratively created graph database for structuring human knowledge" SIGMOD 2008

Object	Number
Facts	3.1 billion
Entities	195 million
Relations	784 977

Table C.2: Statistics of the Freebase knowledge base at the time of its termination. Most relations (around 81%) appear only once in the knowledge base.

`/m/02mjmr` */topic/notable_for* `/g/125920`
`/g/125920` */c…/notable_for/object* `/gov…/us_president`
`/g/125920` */c…/notable_for/predicate* `/type/object/type`

Here `/m/02mjmr` refers to "Barack Obama," while `/g/125920` is the mediator entity which is used to group together several statements about `/m/02mjmr`.

C.4 MUC-7 TR

The message understanding conferences (MUC) were organized by DARPA in the 1980s and 1990s. The seventh—and last—conference (Chinchor 1998) introduced a relation extraction task called "template relation" (TR). Three relations needed to be extracted: *employee of*, *location of* and *product of*. Both the train set and evaluation set contained 100 articles. The task was very much still in the "template filling" mindset; this can be seen by the following example of extracted fact:

Chinchor, "Overview of MUC-7" MUC 1998

```
<EMPLOYEE_OF-9602040136-5> :=
    PERSON: <ENTITY-9602040136-11>
    ORGANIZATION: <ENTITY-9602040136-1>
<ENTITY-9602040136-11> :=
    ENT_NAME: "Dennis Gillespie"
    ENT_TYPE: PERSON
    ENT_DESCRIPTOR: "Capt."
    / "the commander of Carrier Air Wing 11"
    ENT_CATEGORY: PER_MIL
<ENTITY-9602040136-1> :=
ENT_NAME: "NAVY"
ENT_TYPE: ORGANIZATION
ENT_CATEGORY: ORG_GOVT
```

C.5 New York Times

The New York Times Annotated Corpus (NYT, Sandhaus 2008) was widely used for relation extraction. The full dataset contains 1.8 million articles from 1987 to 2007; however, smaller—and sadly, different—subsets are in use. The subset we use in Chapter 3 was first extracted by Marcheggiani and Titov (2016) and is supposed to be similar—but not identical—to the one of Yao et al. (2011). This NYT subset only contains articles from 2000 to 2007 from which "noisy documents" were filtered out. Semi-structured information such as tables and lists were also removed. The version of the dataset we received from Diego Marcheggiani was already preprocessed, with features listed in Section 3.3.2 already extracted.

Sandhaus, "The New York Times Annotated Corpus" LDC 2008

Marcheggiani and Titov, "Discrete-State Variational Autoencoders for Joint Discovery and Factorization of Relations" TACL 2016

The original dataset can be obtained from the following website:

https://catalog.ldc.upenn.edu/LDC2008T19

At the time of writing, once the license fee is paid, the only way to obtain the subset of Marcheggiani and Titov (2016) and Chapter 3 is through someone with access to this specific subset. This burdensome—and expensive—procedure is one of the reasons for which we introduced T-REx-based alternatives in Chapter 3.

C.6 SemEval 2010 Task 8

SemEval is the international workshop on semantic evaluation, which was started in 1998 (then called Senseval) with the goal of emulating the message understanding conferences (Section C.4). In 2010, eighteen different tasks were evaluated. Task number 8 was relation extraction. SemEval 2010 Task 8 (Hendrickx et al. 2010) therefore refers to the dataset provided at the time of this challenge. It is a supervised relation extraction dataset without entity linking and with non-unique entity reference (Section 2.1.2). Its statistics are listed in Table C.3. All samples were hand-labeled by human annotators with one of 19 relations. These 19 relations are built from 9 base relations, which can appear in both directions (Section 2.1.1.3), plus the *other* relation (Section 2.1.1.1). The 9 base relations in the dataset are:

- *cause–effect*
- *instrument–agency*
- *product–producer*
- *content–container*
- *entity–origin*
- *entity–destination*
- *component–whole*
- *member–collection*
- *message–topic*

SemEval 2010 Task 8 introduced an extensive evaluation system, most of which is described in Section 2.3.1. In particular, the official score of the competition was the half-directed macro-$\overleftrightarrow{F_1}$ (described in Section 2.3.1) which was referred to as "9 + 1-way evaluation taking directionality into account."

Hendrickx et al., "SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals" SemEval 2010

Object	Number
Train samples	8 000
Test samples	2 717
Relations	$2 \times 9 + 1 = 19$

Table C.3: Statistics of the SemEval 2010 Task 8 dataset.

C.7 T-REx

T-REx (Elsahar et al. 2018) is an alignment of Wikipedia with Wikidata. In particular, T-REx uses DBpedia abstracts (Brümmer et al. 2016), that is, the introductory paragraphs of Wikipedia's articles. Its statistics are listed in Table C.4.

In the final dataset, entities are linked using the DBpedia spotlight entity linker (Mendes et al. 2011). Furthermore, indirect entity links are extracted using coreference resolution and a "NoSub Aligner," which assumes that the title of the article is implicitly mentioned by all sentences. Finally, some sequences of words are also linked to relations using exact matches of Wikidata relation names. Both the datasets used in Chapters 3 and 4 only consider entities extracted by the spotlight entity linker (tagged `Wikidata_Spotlight_Entity_Linker`). The two datasets of Chapter 3 were filtered based on the tag of the predicate. SPO only contains samples whose predicate's surface form appears in the sentence (tagged `Wikidata_Property_Linker`), while DS contains all samples with the two entities occurring in the same sentence (in other words, all samples except those tagged `NoSubject-Triple-aligner`).

Elsahar et al., "T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples" LREC 2018

Object	Number
Articles	3 million
Sentences	6.2 million
Facts	11 million
Relations	642

Table C.4: Statistics of the T-REx dataset.

C.8 Wikidata

Wikidata (Vrandečić and Krötzsch 2014) is a knowledge base (Section 1.4) started in 2012. Similar to the other projects of the Wikimedia Foundation, it is a collaborative enterprise; everyone can contribute new facts and entities. The introduction of new relations is made through the consensus

Vrandečić and Krötzsch, "Wikidata: A Free Collaborative Knowledgebase" CACM 2014

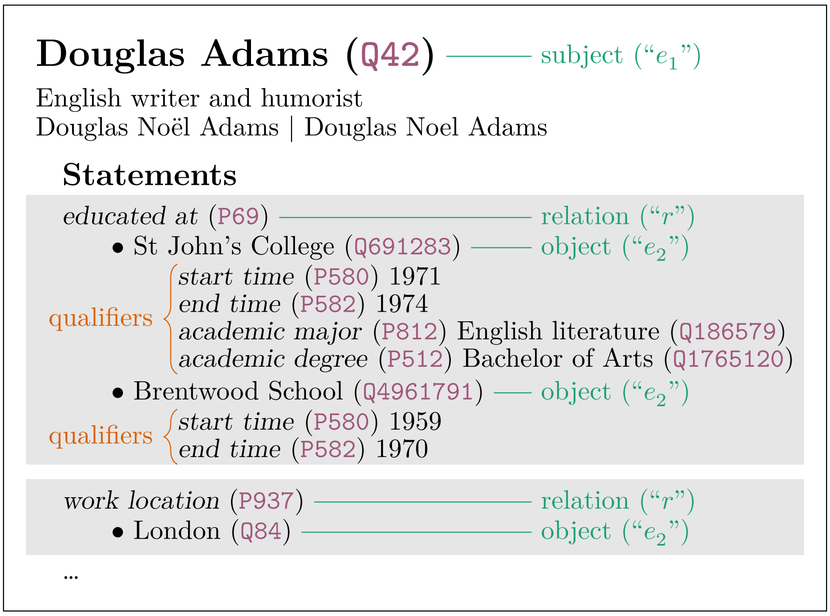


Figure C.1: Structure of a Wikidata page. Facts related to two relations are shown (“statement groups” in Wikidata parlance). This page can be translated into three $\mathcal{E}^2 \times \mathcal{R}$ facts; the first has four additional qualifiers and the second has two additional qualifiers.

of long-term contributors to avoid the explosion of relations types observed on Freebase (section C.3).

Contrary to the way knowledge bases are presented in Section 1.4, Wikidata is not structured as a set of $\mathcal{E}^2 \times \mathcal{R}$ triplets. Instead, in Wikidata, all entities have a page that lists facts of which the entity is the subject. These constitute our set $\mathcal{D}_{\text{KB}} \subseteq \mathcal{E}^2 \times \mathcal{R}$. Furthermore, Wikidata facts can be qualified by additional $\mathcal{R} \times \mathcal{E}$ pairs. For example, Douglas Adams was *educated at* St John’s College *until* 1974. This structure is illustrated in Figure C.1. To be more precise, Wikidata could be modeled as a set of qualified facts, where a qualified fact is an element of $\mathcal{E}^2 \times \mathcal{R} \times 2^{\mathcal{R} \times \mathcal{E}}$.

Bibliography

Abney, Steven (1996). “Statistical methods and linguistics”. In: *The balancing act: Combining symbolic and statistical approaches to language*, pp. 1–26.

Abney, Steven P. (1991). “Parsing by chunks”. In: *Principle-based parsing*. Springer, pp. 257–278.

Agichtein, Eugene and Luis Gravano (2000). “Snowball: Extracting Relations from Large Plain-Text Collections”. In: *Proceedings of the Fifth ACM Conference on Digital Libraries*. San Antonio, Texas, USA: Association for Computing Machinery, pp. 85–94. ISBN: 158113231X. DOI: 10.1145/336597.336644. URL: https://dl.acm.org/doi/pdf/10.1145/336597.336644.

Alex, Beatrice, Claire Grover, Barry Haddow, Mijail Kabadjov, Ewan Klein, Michael Matthews, Stuart Roebuck, Richard Tobin, and Xinglong Wang (2008). “Assisted curation: does text mining really help?” In: *Pacific Symposium on Biocomputing*. Vol. 13, pp. 556–567. URL: https://psb.stanford.edu/psb-online/proceedings/psb08/alex.pdf.

Aone, Chinatsu, Lauren Halverson, Tom Hampton, and Mila Ramos-Santacruz (1998). “SRA: Description of the IE^2 System Used for MUC-7”. In: *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 – May 1, 1998*. URL: https://aclanthology.org/M98-1012.

Auer, Sören, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives (Nov. 2008). “DBpedia: A Nucleus for a Web of Open Data”. In: *Proceedings of 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC+ASWC 2007)*, pp. 722–735. DOI: 10.1007/978-3-540-76298-0_52. URL: http://iswc2007.semanticweb.org/papers/715.pdf.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey Hinton (2016). “Layer Normalization”. arXiv: 1607.06450 [stat.ML].

Babai, László (2015). “Graph Isomorphism in Quasipolynomial Time”. arXiv: 1512.03547 [cs.DS].

— (2016). “Graph Isomorphism in Quasipolynomial Time”. arXiv: 1512.03547 [cs.DS].

Bagga, Amit and Breck Baldwin (Aug. 1998). “Entity-Based Cross-Document Coreferencing Using the Vector Space Model”. In: *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*. Montreal, Quebec, Canada: Association for Computational Linguistics, pp. 79–85. DOI: 10.3115/980845.980859. URL: https://aclanthology.org/P98-1012.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). “Neural Machine Translation by Jointly Learning to Align and Translate”. In: *3rd International Conference on Learning Representations (ICLR), Conference Track Proceedings* (May 7–9, 2015). Ed. by Yoshua Bengio and Yann LeCun. San Diego, CA, USA. URL: http://arxiv.org/abs/1409.0473.

Banko, Michele, Michael Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni (2007). “Open Information Extraction from the Web”. In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence*. Hyderabad, India: Morgan Kaufmann Publishers Inc., pp. 2670–2676. URL: https://www.aaai.org/Papers/IJCAI/2007/IJCAI07-429.pdf.

Barrault, Loïc, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, et al. (Nov. 2020). “Findings of the 2020 Conference on Machine Translation (WMT20)”. In: *Proceedings of the Fifth Conference on Machine Translation*. Online: Association for Computational Linguistics, pp. 1–55. URL: https://aclanthology.org/2020.wmt-1.1.

Beckett, Samuel (1955). *Molloy*.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin (Mar. 2003). “A Neural Probabilistic Language Model”. In: *The Journal of Machine Learning Research* 3, pp. 1137–1155. URL: https://www.jmlr.org/papers/volume3/tmp/bengio03a.pdf.

Berant, Jonathan, Andrew Chou, Roy Frostig, and Percy Liang (Oct. 2013). “Semantic Parsing on Freebase from Question-Answer Pairs”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, pp. 1533–1544. URL: https://aclanthology.org/D13-1160.

Berners-Lee, Tim (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. Harper San Francisco.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. DOI: 10.1162/tacl_a_00051. URL: https://www.aclweb.org/anthology/Q17-1010.

Bollacker, Kurt, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor (2008). “Freebase: a collaboratively created graph database for structuring human knowledge”. In: *SIGMOD ’08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. Vancouver, Canada: Association for Computing Machinery, pp. 1247–1250. ISBN: 978-1-60558-102-6. DOI: 10.1145/1376616.1376746. URL: https://dl.acm.org/doi/pdf/10.1145/1376616.1376746.

Bordes, Antoine, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko (2013). “Translating Embeddings for Modeling Multi-relational Data”. In: *Advances in Neural Information Processing Systems*. Ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Vol. 26. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf.

Boulanger, Auguste (1897). “Contribution à l’étude des équations différentielles linéaires et homogènes intégrables algébriquement”. Thèses de doctorat.

Brachman, Ronald (Oct. 1983). “What IS-A Is and Isn’t: An Analysis of Taxonomic Links in Semantic Networks”. In: *Computer* 16.10, pp. 30–36. ISSN: 1558-0814. DOI: 10.1109/MC.1983.1654194. URL: https://doi.ieeecomputersociety.org/10.1109/MC.1983.1654194.

Brin, Sergey (1999). “Extracting Patterns and Relations from the World Wide Web”. In: *The World Wide Web and Databases*. Ed. by Paolo Atzeni, Alberto Mendelzon, and Giansalvatore Mecca. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 172–183. ISBN: 978-3-540-48909-2. URL: http://ilpubs.stanford.edu:8090/421/1/1999-65.pdf.

Brümmer, Martin, Milan Dojchinovski, and Sebastian Hellmann (May 2016). “DBpedia Abstracts: A Large-Scale, Open, Multilingual NLP Training Corpus”. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 3339–3343. URL: https://aclanthology.org/L16-1532.

Bruna, Joan, Wojciech Zaremba, Arthur D. Szlam, and Yann LeCun (2014). “Spectral Networks and Locally Connected Networks on Graphs”. In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1312.6203.

Cai, Jin-Yi, Martin Fürer, and Neil Immerman (1992). “An optimal lower bound on the number of variables for graph identification”. In: *Combinatorica* 12.4, pp. 389–410. URL: https://people.cs.umass.edu/~immerman/pub/opt.pdf.

Callison-Burch, Chris, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar Zaidan (July 2010). “Findings of the 2010 Joint Workshop on Statistical Machine Translation and Metrics for Machine Translation”. In: *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*. Uppsala, Sweden: Association for Computational Linguistics, pp. 17–53. URL: https://aclanthology.org/W10-1703.

Cegłowski, Maciej (2014). *Web Design: The First 100 Years*. URL: https://idlewords.com/talks/web_design_first_100_years.htm.

Chah, Niel (2017). “Freebase-triples: A Methodology for Processing the Freebase Data Dumps”. arXiv: 1712.08707 [cs.DB].

Chen, Jinxiu, Donghong Ji, Chew Lim Tan, and Zhengyu Niu (July 2006). “Relation Extraction Using Label Propagation Based Semi-Supervised Learning”. In: *Proceedings of the 21st International Conference on*

Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics. Sydney, Australia: Association for Computational Linguistics, pp. 129–136. DOI: 10.3115/1220175.1220192. URL: https://aclanthology.org/P06-1017.

Chevalier, Gil (1990). “Frontispice de la Bibliothèque Oucuipienne”.

Chinchor, Nancy A. (1998). “Overview of MUC-7”. In: *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*. URL: https://aclanthology.org/M98-1001.

Cho, Kyunghyun, Bart van Merriënboer, Çağlar Gulçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (Oct. 2014). “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: https://www.aclweb.org/anthology/D14-1179.

Cohen, Amir DN, Shachar Rosenman, and Yoav Goldberg (2021). “Relation Classification as Two-way Span-Prediction”. Under review for ACL 2022. arXiv: 2010.04829 [cs.CL]. URL: https://arxiv.org/abs/2010.04829.

Collobert, Ronan and Jason Weston (2008). “A unified architecture for natural language processing: deep neural networks with multitask learning”. In: ed. by Andrew McCallum and Sam Roweis, pp. 160–167. DOI: 10.1145/1390156.1390177. URL: https://dl.acm.org/doi/pdf/10.1145/1390156.1390177.

Conard, Louis (1926). “Lettre du 16 mai 1843 à sa sœur”. In: *Correspondance de Gustave Flaubert*. Vol. 1, pp. 139–140.

Conneau, Alexis and Guillaume Lample (2019). “Cross-lingual Language Model Pretraining”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf.

Cortes, Corinna and Vladimir Vapnik (1995). “Support-vector networks”. In: *Machine learning* 20.3, pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018.

Craven, Mark and Johan Kumlien (1999). “Constructing biological knowledge bases by extracting information from text sources”. In: *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*. Vol. 1999, pp. 77–86. URL: https://www.aaai.org/Papers/ISMB/1999/ISMB99-010.pdf.

Culotta, Aron and Jeffrey Sorensen (July 2004). “Dependency Tree Kernels for Relation Extraction”. In: *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*. Barcelona, Spain, pp. 423–429. DOI: 10.3115/1218955.1219009. URL: https://aclanthology.org/P04-1054.

Cuturi, Marco (2013). “Sinkhorn Distances: Lightspeed Computation of Optimal Transport”. In: *Advances in Neural Information Processing Systems*. Ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Vol. 26. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf.

Cybenko, George (1989). “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4, pp. 303–314.

Dalton, Jeffrey, Laura Dietz, and James Allan (2014). “Entity Query Feature Expansion Using Knowledge Base Links”. In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR ’14. Gold Coast, Queensland, Australia: ACM, pp. 365–374. ISBN: 978-1-4503-2257-7. DOI: 10.1145/2600428.2609628. URL: http://doi.acm.org/10.1145/2600428.2609628.

Darroch, John Newton and D. Ratcliff (1972). “Generalized Iterative Scaling for Log-Linear Models”. In: *The Annals of Mathematical Statistics* 43.5, pp. 1470–1480. ISSN: 00034851. URL: http://www.jstor.org/stable/2240069.

Defays, Daniel (1977). “An efficient algorithm for a complete link method”. In: *The Computer Journal* 20.4, pp. 364–366.

Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst (2016). “Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett. Vol. 29. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2016/file/04df4d434d481c5bb723be1b6df1ee65-Paper.pdf.

Saussure, Ferdinand de (1916). *Cours de linguistique générale*. French. Ed. by Albert Bally Charles et Sechehaye. Payot.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://www.aclweb.org/anthology/N19-1423.

Dietterich, Thomas G., Richard H. Lathrop, and Tomás Lozano-Pérez (1997). "Solving the multiple instance problem with axis-parallel rectangles". In: *Artificial Intelligence* 89.1, pp. 31–71. ISSN: 0004-3702. DOI: https://doi.org/10.1016/S0004-3702(96)00034-3. URL: https://www.sciencedirect.com/science/article/pii/S0004370296000343.

Doddington, George R, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel (2004). "The automatic content extraction (ACE) program-tasks, data, and evaluation." In: 2.1, pp. 837–840. URL: https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/lrec2004-ace-program.pdf.

Dostert, Leon E (1955). "The Georgetown–IBM experiment". In: *Machine translation of languages*, pp. 124–135.

Downey, Doug, Oren Etzioni, and Stephen Soderland (2005). "A probabilistic model of redundancy in information extraction". In: *Proceedings of the 19th International Joint Conference on Artifical Intelligence*, pp. 1028–1033. URL: https://www.ijcai.org/Proceedings/05/Papers/1390.pdf.

Dumais, Susan T, George W Furnas, Thomas K Landauer, Scott Deerwester, and Richard Harshman (1988). "Using latent semantic analysis to improve access to textual information". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 281–285. DOI: 10.1145/57167.57214. URL: https://dl.acm.org/doi/pdf/10.1145/57167.57214.

Elsahar, Hady, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl (May 2018). "T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA). URL: https://aclanthology.org/L18-1544.

Fraser, Chris (2007). "Language and Ontology in Early Chinese Thought". In: *Philosophy East and West* 57.4, pp. 420–456. ISSN: 00318221, 15291898. URL: http://www.jstor.org/stable/20109423.

Freund, Yoav and Robert E. Schapire (1999). "Large margin classification using the perceptron algorithm". In: *Machine learning* 37.3, pp. 277–296. ISSN: 1573-0565. DOI: 10.1023/A:1007662407062.

Fu, Tsu-Jui, Peng-Hsuan Li, and Wei-Yun Ma (July 2019). "GraphRel: Modeling Text as Relational Graphs for Joint Entity and Relation Extraction". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 1409–1418. DOI: 10.18653/v1/P19-1136. URL: https://aclanthology.org/P19-1136.

Gage, Philip (1994). "A new algorithm for data compression". In: *C Users Journal* 12.2, pp. 23–38.

Gao, Tianyu, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou (Nov. 2019). "FewRel 2.0: Towards More Challenging Few-Shot Relation Classification". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 6250–6255. DOI: 10.18653/v1/D19-1649. URL: https://aclanthology.org/D19-1649.

Gene Ontology Consortium (Jan. 2004). "The Gene Ontology (GO) database and informatics resource". In: *Nucleic Acids Research* 32, pp. D258–D261. ISSN: 0305-1048. DOI: 10.1093/nar/gkh036. URL: https://academic.oup.com/nar/article-pdf/32/suppl%5C_1/D258/7621365/gkh036.pdf.

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio (2011). "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (Apr. 11–13, 2011). Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA, pp. 315–323. URL: http://proceedings.mlr.press/v15/glorot11a.html.

Google (2016). *Freebase Data Dumps*. URL: https://developers.google.com/freebase/data.

Gracia, Jorge and Lloyd Newton (2016). “Medieval Theories of the Categories”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Winter 2016. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/win2016/entries/medieval-categories/.

Greff, Klaus, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber (2017). “LSTM: A Search Space Odyssey”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10, pp. 2222–2232. DOI: 10.1109/TNNLS.2016.2582924.

Greff, Klaus, Sjoerd van Steenkiste, and Jürgen Schmidhuber (2020). “On the Binding Problem in Artificial Neural Networks”. arXiv: 2012.05208 [cs.NE].

Gumbel, Emil Julius (1954). *Statistical Theory of Extreme Values and Some Practical Applications. A Series of Lectures.* US Government Printing Office. URL: https://ntrl.ntis.gov/NTRL/dashboard/searchResults/titleDetail/PB175818.xhtml.

Gutmann, Michael and Aapo Hyvärinen (2010). “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (May 13–15, 2010). Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. JMLR Workshop and Conference Proceedings. Chia Laguna Resort, Sardinia, Italy, pp. 297–304. URL: http://proceedings.mlr.press/v9/gutmann10a.html.

Hamilton, Will, Zhitao Ying, and Jure Leskovec (2017). “Inductive Representation Learning on Large Graphs”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf.

Han, Xu, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun (Oct. 2018). “FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 4803–4809. DOI: 10.18653/v1/D18-1514. URL: https://aclanthology.org/D18-1514.

Hansen, Chad D. (1983). *Language and logic in ancient China*. University of Michigan Press.

Harbsmeier, Christoph (1989). “Marginalia sino-logica”. In: *Understanding the Chinese mind*, pp. 125–166.

Harris, Zellig S. (1954). “Distributional Structure”. In: *WORD* 10.2–3, pp. 146–162. DOI: 10.1080/00437956.1954.11659520.

Hasegawa, Takaaki, Satoshi Sekine, and Ralph Grishman (July 2004). “Discovering Relations among Named Entities from Large Corpora”. In: *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. Barcelona, Spain, pp. 415–422. DOI: 10.3115/1218955.1219008. URL: https://aclanthology.org/P04-1053.

Hearst, Marti A. (1992). “Automatic Acquisition of Hyponyms from Large Text Corpora”. In: *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*. URL: https://aclanthology.org/C92-2082.

Hendrickx, Iris, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz (July 2010). “SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals”. In: *Proceedings of the 5th International Workshop on Semantic Evaluation*. Uppsala, Sweden: Association for Computational Linguistics, pp. 33–38. URL: https://aclanthology.org/S10-1006.

Higgins, Irina, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner (2017). “β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Sy2fzU9gl.

Hinton, Geoffrey E (1986). “Learning distributed representations of concepts”. In: *Proceedings of the eighth annual conference of the cognitive science society*. Vol. 1. Amherst, MA, USA, p. 12. URL: https://www.cs.toronto.edu/~hinton/absps/families.pdf.

Hinton, Geoffrey E., Simon Osindero, and Yee-Whye Teh (July 2006). “A Fast Learning Algorithm for Deep Belief Nets”. In: *Neural Computation* 18.7, pp. 1527–1554. ISSN: 0899-7667. DOI: 10.1162/neco.2006.18.7.1527. URL: https://direct.mit.edu/neco/article/18/7/1527/7065.

Hochreiter, Sepp (Apr. 1998). “The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, pp. 107–116. doi: 10.1142/S0218488598000094.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). “Long Short-Term Memory”. In: *Neural Computation* 9.8, pp. 1735–1780. issn: 0899-7667. doi: 10.1162/neco.1997.9.8.1735. url: https://direct.mit.edu/neco/article/9/8/1735/6109.

Hoffmann, Raphael, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel Weld (June 2011). “Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations”. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, usa: Association for Computational Linguistics, pp. 541–550. url: https://aclanthology.org/P11-1055.

Hu, Xuming, Lijie Wen, Yusong Xu, Chenwei Zhang, and Philip Yu (Nov. 2020). “SelfORE: Self-supervised Relational Feature Learning for Open Relation Extraction”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (emnlp)*. Online: Association for Computational Linguistics, pp. 3673–3682. doi: 10.18653/v1/2020.emnlp-main.299. url: https://aclanthology.org/2020.emnlp-main.299.

Hu, Ziniu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun (2020). “Heterogeneous Graph Transformer”. In: *Proceedings of The Web Conference 2020*. New York, ny, usa: Association for Computing Machinery, pp. 2704–2710. isbn: 9781450370233. doi: 10.1145/3366423.3380027. url: https://dl.acm.org/doi/pdf/10.1145/3366423.3380027.

Hubert, Lawrence and Phipps Arabie (Dec. 1985). “Comparing partitions”. In: *Journal of classification* 2.1, pp. 193–218. issn: 1432-1343. doi: 10.1007/BF01908075. url: https://link.springer.com/content/pdf/10.1007/BF01908075.pdf.

Immerman, Neil and Eric Lander (1990). “Describing Graphs: A First-Order Approach to Graph Canonization”. In: *Complexity Theory Retrospective: In Honor of Juris Hartmanis on the Occasion of His Sixtieth Birthday, July 5, 1988*. Ed. by Alan L. Selman. New York, ny, usa: Springer New York, pp. 59–81. isbn: 978-1-4612-4478-3. doi: 10.1007/978-1-4612-4478-3_5. url: https://www.cs.yale.edu/publications/techreports/tr605.pdf.

Jang, Eric, Shixiang Gu, and Ben Poole (2016). “Categorical reparameterization with gumbel–softmax”. In: *International Conference on Learning Representations*. url: https://openreview.net/forum?id=rkE3y85ee.

Jarry, Alfred (1911). *Gestes et opinions du docteur Faustroll*.

Jiang, Tianwen, Sendong Zhao, Jing Liu, Jin-Ge Yao, Ming Liu, Bing Qin, Ting Liu, and Chin-Yew Lin (2019). “Towards Time-Aware Distant Supervision for Relation Extraction”. arXiv: 1903.03289 [cs.CL].

Jozefowicz, Rafal, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu (2016). “Exploring the Limits of Language Modeling”. arXiv: 1602.02410 [cs.CL].

Kambhatla, Nanda (July 2004). “Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Information Extraction”. In: *Proceedings of the acl Interactive Poster and Demonstration Sessions*. Barcelona, Spain: Association for Computational Linguistics, pp. 178–181. url: https://aclanthology.org/P04-3022.

Kim, Yoon (Oct. 2014). “Convolutional Neural Networks for Sentence Classification”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (emnlp)*. Doha, Qatar: Association for Computational Linguistics, pp. 1746–1751. doi: 10.3115/v1/D14-1181. url: https://www.aclweb.org/anthology/D14-1181.

Kingma, Diederik P. and Max Welling (2014). “Auto-Encoding Variational Bayes”. In: *2nd International Conference on Learning Representations, iclr 2014, Banff, ab, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. url: http://arxiv.org/abs/1312.6114.

Kipf, Thomas N and Max Welling (2017). “Semi-Supervised Classification with Graph Convolutional Networks”. In: *International Conference on Learning Representations*. url: https://openreview.net/forum?id=SJU4ayYgl.

Klein, Dan and Christopher Manning (July 2003). “Accurate Unlexicalized Parsing”. In: *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*. Sapporo, Japan: Association for

Computational Linguistics, pp. 423–430. DOI: 10.3115/1075096.1075150. URL: https://aclanthology.org/P03-1054.
Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Vol. 25. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.
LeCun, Yann and Ishan Misra (Mar. 4, 2021). *Self-supervised learning: The dark matter of intelligence*. URL: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence (visited on 11/08/2021).
Lee, Jinhyuk, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang (Sept. 2019). "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4, pp. 1234–1240. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz682. URL: https://academic.oup.com/bioinformatics/article-pdf/36/4/1234/32527770/btz682.pdf.
Leshno, Moshe, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken (1993). "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural networks* 6.6, pp. 861–867.
Levy, Omer and Yoav Goldberg (2014). "Neural Word Embedding as Implicit Matrix Factorization". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger. Vol. 27. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2014/file/feab05aa91085b7a8012516bc3533958-Paper.pdf.
Lin, Dekang and Patrick Pantel (2001). "DIRT – Discovery of Inference Rules from Text". In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, California: Association for Computing Machinery, pp. 323–328. ISBN: 158113391X. DOI: 10.1145/502512.502559. URL: http://www.patrickpantel.com/download/papers/2001/kdd01-1.pdf.
Lin, Yankai, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu (2015). "Learning Entity and Relation Embeddings for Knowledge Graph Completion". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. Austin, Texas: AAAI Press, pp. 2181–2187. ISBN: 0262511290.
Lin, Yankai, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun (Aug. 2016). "Neural Relation Extraction with Selective Attention over Instances". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 2124–2133. DOI: 10.18653/v1/P16-1200. URL: https://aclanthology.org/P16-1200.
Maas, Andrew L, Awni Y Hannun, Andrew Y Ng, et al. (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. Vol. 30. 1, p. 3. URL: https://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf.
Marcheggiani, Diego and Ivan Titov (2016). "Discrete-State Variational Autoencoders for Joint Discovery and Factorization of Relations". In: *Transactions of the Association for Computational Linguistics* 4, pp. 231–244. DOI: 10.1162/tacl_a_00095. URL: https://aclanthology.org/Q16-1017.
Marque-Pucheu, Christiane (2008). "La couleur des prépositions à et de". In: vol. 157. Paris, France: Armand Colin, pp. 74–105. DOI: 10.3917/lf.157.0074. URL: https://www.cairn.info/load_pdf.php?ID_ARTICLE=LF_157_0074.
Mathon, Rudolf (1979). "A note on the graph isomorphism counting problem". In: *Information Processing Letters* 8.3, pp. 131–136.
McCann, Bryan, James Bradbury, Caiming Xiong, and Richard Socher (2017). "Learned in Translation: Contextualized Word Vectors". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/20c86a628232a67e7bd46f76fba7ce12-Paper.pdf.
McCarthy, John (1959). "Programs with common sense". In: URL: http://www-formal.stanford.edu/jmc/mcc59/mcc59.html.
McDonald, Ryan, Fernando Pereira, Seth Kulick, Scott Winters, Yang Jin, and Pete White (June 2005). "Simple Algorithms for Complex Relation Extraction with Applications to Biomedical IE". In: *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Ann Arbor,

Michigan: Association for Computational Linguistics, pp. 491–498. doi: 10.3115/1219840.1219901. url: https://aclanthology.org/P05-1061.

Mendes, Pablo N., Max Jakob, Andrés García-Silva, and Christian Bizer (2011). "dbpedia Spotlight: Shedding Light on the Web of Documents". In: *Proceedings of the 7th International Conference on Semantic Systems*. I-Semantics '11. Graz, Austria: Association for Computing Machinery, pp. 1–8. isbn: 9781450306218. doi: 10.1145/2063518.2063519. url: https://dl.acm.org/doi/pdf/10.1145/2063518.2063519.

Mesquita, Filipe, Matteo Cannaviccio, Jordan Schmidek, Paramita Mirza, and Denilson Barbosa (Nov. 2019). "KnowledgeNet: A Benchmark Dataset for Knowledge Base Population". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (emnlp-ijcnlp)*. Hong Kong, China: Association for Computational Linguistics, pp. 749–758. doi: 10.18653/v1/D19-1069. url: https://aclanthology.org/D19-1069.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013a). "Efficient Estimation of Word Representations in Vector Space". arXiv: 1301.3781 [cs.CL].

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013b). "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems*. Ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Vol. 26. Curran Associates, Inc. url: https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

Miller, George A. (Nov. 1995). "WordNet: A Lexical Database for English". In: *Communications of the acm* 38.11, pp. 39–41. issn: 0001-0782. doi: 10.1145/219717.219748.

Miller, Scott, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, and The Annotation Group (1998). "bbn: Description of the sift System as Used for muc-7". In: *Seventh Message Understanding Conference (muc-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 – May 1, 1998*. url: https://aclanthology.org/M98-1009.

Mintz, Mike, Steven Bills, Rion Snow, and Daniel Jurafsky (Aug. 2009). "Distant supervision for relation extraction without labeled data". In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the acl and the 4th International Joint Conference on Natural Language Processing of the afnlp*. Suntec, Singapore: Association for Computational Linguistics, pp. 1003–1011. url: https://aclanthology.org/P09-1113.

Mnih, Andriy and Yee Whye Teh (2012). "A fast and simple algorithm for training neural probabilistic language models". In: *Proceedings of the 29th International Conference on Machine Learning*, p. 58. url: http://icml.cc/2012/papers/855.pdf.

Montariol, Syrielle, Étienne Simon, Arij Riabi, and Djamé Seddah (May 2022). "Fine-tuning and Sampling Strategies for Multimodal Role Labeling of Entities under Class Imbalance". In: *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*. Dublin, Ireland: Association for Computational Linguistics, pp. 55–65. url: https://aclanthology.org/2022.constraint-1.7.

Morgan, Augustus De (1864). "On the Syllogism, No. III, and on Logic in general". In: *Transactions of the Cambridge Philosophical Society* 10, pp. 173–230.

Morris, Christopher, Nils M. Kriege, Kristian Kersting, and Petra Mutzel (2016). "Faster Kernels for Graphs with Continuous Attributes via Hashing". In: *2016 ieee 16th International Conference on Data Mining (icdm)*. ieee, pp. 1095–1100. doi: 10.1109/ICDM.2016.0142.

Morris, Christopher, Gaurav Rattan, and Petra Mutzel (2020). "Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 21824–21840. url: https://proceedings.neurips.cc/paper/2020/file/f81dee42585b3814de199b2e88757f5c-Paper.pdf.

Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel (June 2011). "A Three-Way Model for Collective Learning on Multi-Relational Data". In: *Proceedings of the 28th International Conference on Machine Learning (icml-11)*. Ed. by Lise Getoor and Tobias Scheffer. Bellevue, wa, usa: acm, pp. 809–816. isbn: 978-1-4503-0619-5. url: https://icml.cc/2011/papers/438_icmlpaper.pdf.

Norvig, Peter (2011). *On Chomsky and the Two Cultures of Statistical Learning*. url: https://norvig.com/chomsky.html.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://www.aclweb.org/anthology/D14-1162.

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena (2014). “DeepWalk: Online Learning of Social Representations”. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, pp. 701–710. ISBN: 9781450329569. DOI: 10.1145/2623330.2623732. URL: https://dl.acm.org/doi/pdf/10.1145/2623330.2623732.

Peters, Matthew, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (June 2018). “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: https://www.aclweb.org/anthology/N18-1202.

Poincaré, Henri (1908). *Thermodynamique*. Gauthier-Villars.

Qian, Yujie, Enrico Santus, Zhijing Jin, Jiang Guo, and Regina Barzilay (June 2019). “GraphIE: A Graph-Based Framework for Information Extraction”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 751–761. DOI: 10.18653/v1/N19-1082. URL: https://aclanthology.org/N19-1082.

Qu, Meng, Tianyu Gao, Louis-Pascal Xhonneux, and Jian Tang (July 2020). “Few-shot Relation Extraction via Bayesian Meta-learning on Relation Graphs”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 7867–7876. URL: https://proceedings.mlr.press/v119/qu20a.html.

Quine, Willard Van Orman (1951). “Main Trends in Recent Philosophy: Two Dogmas of Empiricism”. In: *The Philosophical Review* 60.1, pp. 20–43. ISSN: 00318108, 15581470. URL: http://www.jstor.org/stable/2181906.

— (2004). *Du point de vue logique : neuf essais logico-philosophiques*. Trans. by Sandra Laugier. Vrin.

Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever (2018). “Improving Language Understanding by Generative Pre-Training”.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu (2020). “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *Journal of Machine Learning Research* 21.140, pp. 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

Rand, William M. (1971). “Objective Criteria for the Evaluation of Clustering Methods”. In: *Journal of the American Statistical Association* 66.336, pp. 846–850. DOI: 10.1080/01621459.1971.10482356.

Redouté, Pierre-Joseph (1802). “Paris Quadrifolia”. In: *Les Liliacées*. URL: https://commons.wikimedia.org/wiki/File:Paris_quadrifolia_in_Les_liliacees.jpg. Via Wikimedia Commons.

Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme (2009). “BPR: Bayesian Personalized Ranking from Implicit Feedback”. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. Montreal, Quebec, Canada: AUAI Press, pp. 452–461. ISBN: 9780974903958. DOI: 10.5555/1795114.1795167. URL: https://dl.acm.org/doi/pdf/10.5555/1795114.1795167.

Riedel, Sebastian, Limin Yao, Andrew McCallum, and Benjamin Marlin (June 2013). “Relation Extraction with Matrix Factorization and Universal Schemas”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, pp. 74–84. URL: https://aclanthology.org/N13-1008.

Roberts, Ben and Dirk P Kroese (2007). “Estimating the Number of s–t Paths in a Graph.” In: *Journal of Graph Algorithms and Applications* 11.1, pp. 195–214.

Rosenberg, Andrew and Julia Hirschberg (June 2007). “V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure”. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic: Association for Computational Linguistics, pp. 410–420. URL: https://aclanthology.org/D07-1043.

Sager, Naomi (1972). “Syntactic Formatting of Science Information”. In: *Proceedings of the December 5-7, 1972, Fall Joint Computer Conference, Part II.* Anaheim, California: Association for Computing Machinery, pp. 791–800. ISBN: 9781450379137. DOI: 10.1145/1480083.1480101. URL: https://dl.acm.org/doi/pdf/10.1145/1480083.1480101.

Sandhaus, Evan (2008). *The New York Times Annotated Corpus.* LDC2008T19. Philadelphia: Linguistic Data Consortium. DOI: 10.35111/77ba-9x74. URL: https://catalog.ldc.upenn.edu/LDC2008T19.

Schlichtkrull, Michael, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling (2018). “Modeling Relational Data with Graph Convolutional Networks”. In: *The Semantic Web.* Ed. by Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam. Cham: Springer International Publishing, pp. 593–607. ISBN: 978-3-319-93417-4. URL: https://arxiv.org/pdf/1703.06103.pdf.

Shuman, David I, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst (2013). “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains”. In: *IEEE Signal Processing Magazine* 30.3, pp. 83–98. DOI: 10.1109/MSP.2012.2235192. URL: https://arxiv.org/pdf/1211.0053.pdf.

Simon, Étienne, Vincent Guigue, and Benjamin Piwowarski (July 2019). “Unsupervised Information Extraction: Regularizing Discriminative Approaches with Relation Distribution Losses”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* Florence, Italy: Association for Computational Linguistics, pp. 1378–1387. DOI: 10.18653/v1/P19-1133. URL: https://www.aclweb.org/anthology/P19-1133.

Soames, Scott (1997). “Skepticism about Meaning: Indeterminacy, Normativity, and the Rule-Following Paradox”. In: *Canadian Journal of Philosophy Supplementary Volume* 23, pp. 211–249. DOI: 10.1080/00455091.1997.10715967.

Soares, Livio Baldini, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski (July 2019). “Matching the Blanks: Distributional Similarity for Relation Learning”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* Florence, Italy: Association for Computational Linguistics, pp. 2895–2905. DOI: 10.18653/v1/P19-1279. URL: https://aclanthology.org/P19-1279.

Socher, Richard, Danqi Chen, Christopher D Manning, and Andrew Ng (2013). “Reasoning With Neural Tensor Networks for Knowledge Base Completion”. In: *Advances in Neural Information Processing Systems.* Ed. by C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger. Vol. 26. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2013/file/b337e84de8752b27eda3a12363109e80-Paper.pdf.

Socher, Richard, Brody Huval, Christopher D. Manning, and Andrew Y. Ng (July 2012). “Semantic Compositionality through Recursive Matrix-Vector Spaces”. In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.* Jeju Island, Korea: Association for Computational Linguistics, pp. 1201–1211. URL: https://aclanthology.org/D12-1110.

Sohn, Kihyuk, Honglak Lee, and Xinchen Yan (2015). “Learning Structured Output Representation using Deep Conditional Generative Models”. In: *Advances in Neural Information Processing Systems.* Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

Song, Linfeng, Yue Zhang, Zhiguo Wang, and Daniel Gildea (Oct. 2018). “N-ary Relation Extraction using Graph-State LSTM”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.* Brussels, Belgium: Association for Computational Linguistics, pp. 2226–2235. DOI: 10.18653/v1/D18-1246. URL: https://aclanthology.org/D18-1246.

Speaks, Jeff (2021). “Theories of Meaning”. In: *The Stanford Encyclopedia of Philosophy.* Ed. by Edward N. Zalta. Spring 2021. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/spr2021/entries/meaning/.

Sperduti, A. and A. Starita (1997). “Supervised neural networks for the classification of structures”. In: *IEEE Transactions on Neural Networks* 8.3, pp. 714–735. DOI: 10.1109/72.572108.

Suárez, Jorge A (1983). *The mesoamerican indian languages.* Cambridge University Press.

Sukhbaatar, Sainbayar, Arthur Szlam, Jason Weston, and Rob Fergus (2015). “End-To-End Memory Networks”. In: *Advances in Neural Information Processing Systems.* Ed. by C. Cortes, N. Lawrence, D. Lee,

M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2015/file/8fb21ee7a2207526da55a679f0332de2-Paper.pdf.

Surdeanu, Mihai, Julie Tibshirani, Ramesh Nallapati, and Christopher Manning (July 2012). "Multi-instance Multi-label Learning for Relation Extraction". In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Jeju Island, Korea: Association for Computational Linguistics, pp. 455–465. URL: https://aclanthology.org/D12-1042.

Sutskever, Ilya, James Martens, and Geoffrey Hinton (June 2011). "Generating Text with Recurrent Neural Networks". In: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. Ed. by Lise Getoor and Tobias Scheffer. Bellevue, Washington, USA: Association for Computing Machinery, pp. 1017–1024. ISBN: 978-1-4503-0619-5.

Tang, Lei and Huan Liu (2009). "Relational Learning via Latent Social Dimensions". In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. Paris, France: Association for Computing Machinery, pp. 817–826. ISBN: 9781605584959. DOI: 10.1145/1557019.1557109. URL: https://dl.acm.org/doi/pdf/10.1145/1557019.1557109.

Tenniel, John (1889). "Cheshire Cat details from the Tree Above Alice". In: *The Nursery "Alice"*. URL: https://commons.wikimedia.org/wiki/File:Tennel_Cheshire_proof.png. Via Wikimedia Commons.

British Museum, the (100 BCE–100 CE). "Ariadne waking on the shore of Naxos". URL: https://www.britishmuseum.org/collection/image/254690001. Wall painting from Herculaneum, Asset number: 254690001, Museum number: 1867,0508.1358.

Togninalli, Matteo, Elisabetta Ghisu, Felipe Llinares-López, Bastian Rieck, and Karsten Borgwardt (2019). "Wasserstein Weisfeiler-Lehman Graph Kernels". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/73fed7fd472e502d8908794430511f4d-Paper.pdf.

Trisedya, Bayu Distiawan, Gerhard Weikum, Jianzhong Qi, and Rui Zhang (July 2019). "Neural Relation Extraction for Knowledge Base Enrichment". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 229–240. DOI: 10.18653/v1/P19-1023. URL: https://aclanthology.org/P19-1023.

Turing, Alan Mathison (Oct. 1950). "Computing Machinery and Intelligence". In: *Mind* LIX.236, pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. URL: https://academic.oup.com/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf.

Tyler, Andrea and Vyvyan Evans (2001). "Reconsidering prepositional polysemy networks: The case of over". In: *Language*, pp. 724–765.

Ushio, Asahi, Jose Camacho-Collados, and Steven Schockaert (Nov. 2021). "Distilling Relation Embeddings from Pretrained Language Models". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 9044–9062. DOI: 10.18653/v1/2021.emnlp-main.712. URL: https://aclanthology.org/2021.emnlp-main.712.

Valiant, Leslie G. (1979). "The Complexity of Enumeration and Reliability Problems". In: *SIAM Journal on Computing* 8.3, pp. 410–421. DOI: 10.1137/0208032.

Oord, Aäron van den, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu (2016). "WaveNet: A Generative Model for Raw Audio". arXiv: 1609.03499 [cs.SD].

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio (2018). "Graph Attention Networks". In: URL: https://openreview.net/forum?id=rJXMpikCZ.

Vincent, Pascal, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol (2010). "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion”. In: *Journal of Machine Learning Research* 11.110, pp. 3371–3408. URL: http://jmlr.org/papers/v11/vincent10a.html.

Vrandečić, Denny and Markus Krötzsch (Sept. 2014). “Wikidata: A Free Collaborative Knowledgebase”. In: *Communications of the ACM* 57.10, pp. 78–85. ISSN: 0001-0782. DOI: 10.1145/2629489. URL: https://dl.acm.org/doi/pdf/10.1145/2629489.

Waibel, Alex, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang (1989). “Phoneme recognition using time-delay neural networks”. In: *IEEE transactions on acoustics, speech, and signal processing* 37.3, pp. 328–339.

Wang, Xiao, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu (2019). “Heterogeneous Graph Attention Network”. In: *The World Wide Web Conference*. San Francisco, CA, USA: Association for Computing Machinery, pp. 2022–2032. ISBN: 9781450366748. DOI: 10.1145/3308558.3313562. URL: https://dl.acm.org/doi/pdf/10.1145/3308558.3313562.

Wang, Zhen, Jianwen Zhang, Jianlin Feng, and Zheng Chen (2014). “Knowledge Graph Embedding by Translating on Hyperplanes”. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. AAAI’14. Québec City, Québec, Canada: AAAI Press, pp. 1112–1119.

Watterson, Bill (May 17, 1992). *Calvin and Hobbes*.

Weisfeiler, Boris and Andreĭ Leman (1968). “The reduction of a graph to canonical form and the algebra which appears therein”. In: *NTI, Series* 2.9, pp. 12–16. URL: https://www.iti.zcu.cz/wl2018/pdf/wl_paper_translation.pdf.

Weston, Jason, Sumit Chopra, and Antoine Bordes (2015). “Memory Networks”. In: *3rd International Conference on Learning Representations (ICLR), Conference Track Proceedings* (May 7–9, 2015). Ed. by Yoshua Bengio and Yann LeCun. San Diego, CA, USA. URL: http://arxiv.org/abs/1410.3916.

Xie, Junyuan, Ross Girshick, and Ali Farhadi (June 2016). “Unsupervised Deep Embedding for Clustering Analysis”. In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, pp. 478–487. URL: https://proceedings.mlr.press/v48/xieb16.html.

Yamaguchi, Kouichi, Kenji Sakamoto, and Toshio Akabane (Nov. 1990). “A neural network for speaker-independent isolated word recognition”. In: First International Conference on Spoken Language Processing. Kobe, Japan, pp. 1077–1080. URL: https://www.isca-speech.org/archive/icslp_1990/i90_1077.html.

Yang, Zhilin, William Cohen, and Ruslan Salakhudinov (June 2016). “Revisiting Semi-Supervised Learning with Graph Embeddings”. In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, NY, USA: PMLR, pp. 40–48. URL: https://proceedings.mlr.press/v48/yanga16.html.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). “XLNet: Generalized Autoregressive Pretraining for Language Understanding”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf.

Yao, Limin, Aria Haghighi, Sebastian Riedel, and Andrew McCallum (July 2011). “Structured Relation Discovery using Generative Models”. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK: Association for Computational Linguistics, pp. 1456–1466. URL: https://aclanthology.org/D11-1135.

Yao, Limin, Sebastian Riedel, and Andrew McCallum (July 2012). “Unsupervised Relation Discovery with Sense Disambiguation”. In: Jeju Island, Korea: Association for Computational Linguistics, pp. 712–720. URL: https://aclanthology.org/P12-1075.

Yates, Alexander, Michele Banko, Matthew Broadhead, Michael Cafarella, Oren Etzioni, and Stephen Soderland (Apr. 2007). “TextRunner: Open Information Extraction on the Web”. In: *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*. Rochester, NY, USA: Association for Computational Linguistics, pp. 25–26. URL: https://aclanthology.org/N07-4013.

Yates, Alexander and Oren Etzioni (Apr. 2007). “Unsupervised Resolution of Objects and Relations on the Web”. In: *Human Language Technologies 2007: The Conference of the North American Chapter of the*

Association for Computational Linguistics; Proceedings of the Main Conference. Rochester, New York: Association for Computational Linguistics, pp. 121–130. URL: https://aclanthology.org/N07-1016.

Yih, Wen-tau, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao (2015). "Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, pp. 1321–1331. DOI: 10.3115/v1/P15-1128. URL: http://aclweb.org/anthology/P15-1128.

Yuan, Chenhan and Hoda Eldardiry (Nov. 2021). "Unsupervised Relation Extraction: A Variational Autoencoder Approach". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 1929–1938. DOI: 10.18653/v1/2021.emnlp-main.147. URL: https://aclanthology.org/2021.emnlp-main.147.

Zelenko, Dmitry, Chinatsu Aone, and Anthony Richardella (Mar. 2003). "Kernel Methods for Relation Extraction". In: *The Journal of Machine Learning Research* 3, pp. 1083–1106. ISSN: 1532-4435. URL: https://www.jmlr.org/papers/volume3/zelenko03a/zelenko03a.pdf.

Zemlyachenko, Viktor N, Nickolay M Korneenko, and Regina I Tyshkevich (1985). "Graph isomorphism problem". In: *Journal of Soviet Mathematics* 29.4, pp. 1426–1481.

Zeng, Daojian, Kang Liu, Yubo Chen, and Jun Zhao (Sept. 2015). "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 1753–1762. DOI: 10.18653/v1/D15-1203. URL: https://aclanthology.org/D15-1203.

Zhao, Yi, Huaiyu Wan, Jianwei Gao, and Youfang Lin (Nov. 2019). "Improving Relation Classification by Entity Pair Graph". In: *Proceedings of The Eleventh Asian Conference on Machine Learning*. Ed. by Wee Sun Lee and Taiji Suzuki. Vol. 101. Proceedings of Machine Learning Research, pp. 1156–1171. URL: https://proceedings.mlr.press/v101/zhao19a.html.

Zhou, GuoDong, Jian Su, Jie Zhang, and Min Zhang (June 2005). "Exploring Various Knowledge in Relation Extraction". In: *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Ann Arbor, Michigan: Association for Computational Linguistics, pp. 427–434. DOI: 10.3115/1219840.1219893. URL: https://aclanthology.org/P05-1053.

Zhu, Hao, Yankai Lin, Zhiyuan Liu, Jie Fu, Tat-Seng Chua, and Maosong Sun (July 2019). "Graph Neural Networks with Generated Parameters for Relation Extraction". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 1331–1339. DOI: 10.18653/v1/P19-1128. URL: https://aclanthology.org/P19-1128.

Zhu, Xiaojin and Zoubin Ghahramani (2002). "Learning from labeled and unlabeled data with label propagation". In: *Technical Report CMU-CALD*. URL: https://mlg.eng.cam.ac.uk/zoubin/papers/CMU-CALD-02-107.pdf.

Colophon

This document is written in LuaLaTeX using PGF/Ti*k*Z and PGFPLOTS for figures. Most of the text and math are typeset in Latin Modern, while EB Garamond is used for titles. A small amount of characters are from the TeX Gyre Bonum and XITS fonts. Greek words are typeset in the Greek Font Society's Didot Classic, while Chinese excerpts are in the I.Ming font. Finally, the word "THÈSE" on the title page comes from a vectorization of Auguste Boulanger's Ph.D. theses (1897).

The manuscript and sources are freely available at https://esimon.eu/PhD.