UPMC
SORBONNE UNIVERSITÉS

M I 0 2 4

Projet IAD

Vers une Machine d'apprentissage persistante

Cahier des charges

Auteur :
Étienne Simon

Encadrant :
Ludovic Denoyer

5 juillet 2014

Table des matières

1 Objectif du PIAD

1.1 Présentation

Ce PIAD est intitulé « Vers une Machine d'apprentissage persistante. » L'objectif sera d'écrire une plateforme pour l'apprentissage et le raisonnement sur des données hétérogènes. Les modèles ainsi appris devront être conservés dans une base de données en vue de leur réutilisation.

1.2 Cadre

Le projet sera utilisé dans le cadre de la classification dans des réseaux hétérogènes. Cela fera en projetant les données de différents types dans un espace latente (commun).

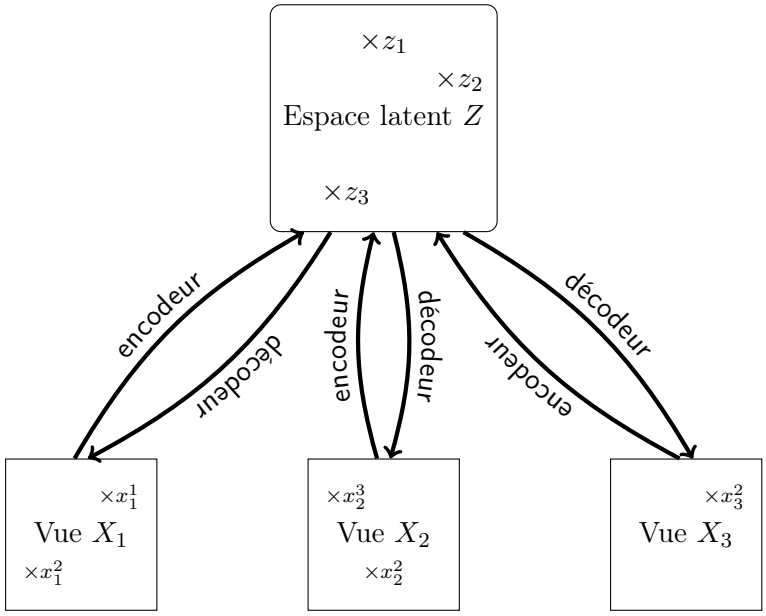

FIGURE 1 – Relation entre les différents espaces

La figure 1 présente les différents espaces entrant en jeu :

- Les vues sont les espaces d'observation, il y en a autant que de type de donnée. Par exemple la vue 1 peut être un espace d'observation textuel et la vue 2 un espace d'observation visuel, au quel cas le vecteur x_1^2 correspond à un mot du type « chat » et le vecteur x_2^2 à l'image d'un chat.
- L'espace latent contient les concepts associés aux observations. En conservant l'exemple précèdent, z_2 correspond au concept « chat. »

Pour faire le lien entre les vecteurs des espaces d'observation et le vecteur de l'espace latent, des encodeur et décodeur sont utilisés, il s'agit de fonctions

$e_i : X_i \mapsto Z$ et $d_i : Z \mapsto X_i$. Celles-ci sont apprises par des perceptrons multicouches (MLP) selon les observations faites.

Par ailleurs, des relations sont définis sur les concepts. Les relations sont des sous ensemble de Z^2 elles aussi apprises par des MLP (qui apprendront plutôt des métriques du type $Z^2 \mapsto \mathbb{R}$.) Un exemple de relation est « auteur de » dans le cadre d'un réseau avec des nœuds « article » et « personne. »

1.3 Objectifs principaux

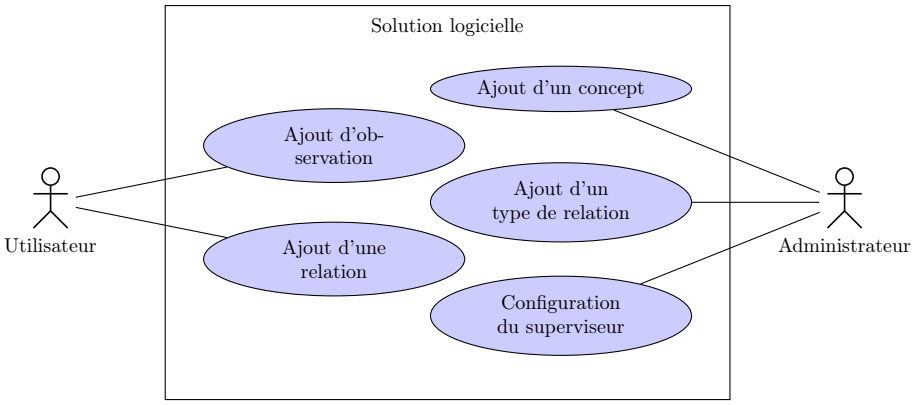

FIGURE 2 – Diagramme des cas d'utilisation

L'objectif principal est d'écrire une plateforme d'apprentissage de représentations latentes. Les objets manipulés doivent être persistant, ainsi, on peut distinguer deux bases de données :

- Une base contenant les données d'apprentissage.
- Une base contenant les modèles appris.

Les modèles sont appris sous l'égide d'un superviseur, se sont des MLP et il sont décomposables en trois catégories :

- Les encodeurs qui permettent de projeter un vecteur observé dans l'espace latent.
- Les décodeurs qui permettent de projeter un vecteur concept de l'espace latent dans un espace d'observation.
- Les métriques qui permettent d'établir des relations entre les concepts de l'espace latent.

1.4 Contraintes techniques

NMLP est une bibliothèque permettant l'apprentissage de MLP. Les MLP sont traités comme des ensembles de modules, qui sont appris sous la gouvernance d'un critère. La plateforme développée utilisera NMLP pour la représentation et l'apprentissage des modèles.

Le développement se fera en C++03 (ISO/IEC 14882 :2003). De plus, le livrable devra se limiter autant que possible aux dépendances de NMLP, c'est à dire les bibliothèques Boost et CUDA. Par ailleurs, le code devra fonctionner sous Windows et sous les divers *nix, n'ayant à disposition que des environnements FreeBSD et OpenBSD avec GCC et Clang, l'encadrant devra se charger de vérifier le fonctionnement du code sous Windows. Le livrable devra être compilable avec le moteur de production CMake utilisé par NMLP.

Pour s'assurer d'un suivi en temps réel du travail effectué, l'ensemble du livrable sera maintenue par un logiciel de gestion de versions, en l'occurrence un dépôt Mercurial sur Bitbucket :

`http://hg.esimon.eu/mi024`

2 Description de la solution demandée

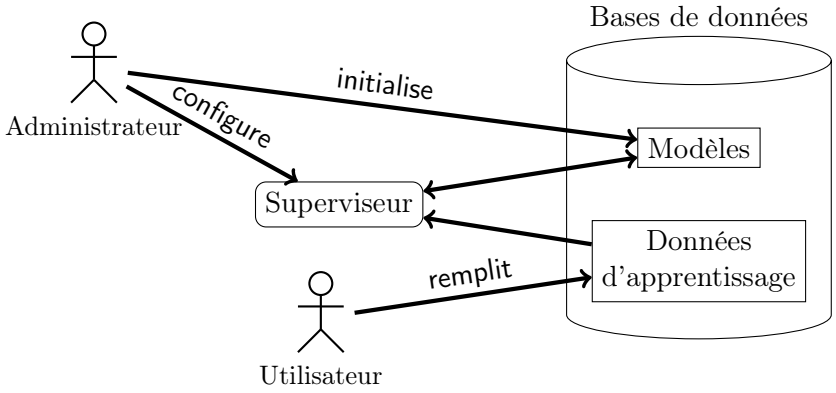

FIGURE 3 – Diagramme (informel) présentant l'architecture de la solution logicielle.

La figure 3 présente les principaux composant de l'architecture logicielle. La majeure partie du travail s'effectuera sur le superviseur. À noté, qu'une fonctionnalité intéressante qui pourra être ajouté est le lancement en parallèle de plusieurs superviseurs (chacun sur un modèle différent), il pourra alors être intéressant de construire un composant pour gérer un ensemble de superviseur (qui enverrait les mêmes données à des superviseurs paramétrés différemment par exemple.)

Une description plus détaillée avec une liste des activités et un calendrier sont présents dans le plan de développement.

3 Composition du livrable

– Un script de compilation CMake

- Des binaires superviseurs
- Une interface d'ajout de données d'apprentissage
- Un contrôleur (serveur) gérant un ensemble de superviseur
- La documentation du projet
  - Le cahier des charges
  - Le plan de développement
  - Le dossier d'analyse et de conception
  - La documentation développeur
  - Le rapport des tests
  - Les manpages relatives aux différents binaires
  - Les README et INSTALL usuels
  - Le rapport final