Rapport du TER GMIN401 :
Intégration et optimisation d'algorithmes de classifications supervisées pour Weka

Par : ALIJATE Mehdi - NEGROS Hadrien - TURKI Batoul

31 Janvier 2014

Table des matières

Résumé

Ce sujet vise à intégrer et à optimiser des algorithmes de classifications supervisées de documents dans la suite logiciel WEKA. Ces algorithmes sont issus de travaux de recherche menés récemment au sein du LIRMM.

1 Introduction

La classification de documents est le mécanisme consistant à classer automatiquement des ressources la classe prédéfinie lui correspondant le mieux.
Plusieurs formes de classification existent (par genre, par opinion, par thème...etc), et se font via des algorithmes de classifications spécifiques. Ceux-ci se basent sur des méthodes principalement numériques (probabilistes), avec des algorithmes utilisant les mathématiques ou basés sur la recherche d'information.
Ce TER vise justement à intégrer des algorithmes de classifications supervisées de documents dans la suite logiciel WEKA [1], se basant sur un nouveau modèle de classification à partir d'un faible nombre de document, intégrant de nouvelles pondérations adaptées.
Tout d'abord, il faudra explorer l'API de WEKA, pour prendre en main du code source, la maniabilité des classes et explorer une méthode d'ajout d'un algorithme de classification. Ensuite, nous nous pencherons sur le développement des différentes classes en établissant une méthodologie concrétisant le travail mené au laboratoire du LIRMM, s'en suivra une phase d'intégration et de tests.

Ce présent mini-rapport présente un compte rendu de la première phase de notre travail, qui s'est déroulée entre notre dernière réunion le 24/01/14 et aujourd'hui. Il sera en partie intégré au rapport final.

2 Exploration de WEKA

Après la réunion du 24/01/14, nous avons établi un plan de travail pour bien mener et répartir les tâches de ce TER. Il a été décidé de le diviser en trois grandes parties. La première, qui est décrite ci-dessous consiste à explorer et prendre en main l'API de WEKA, afin de pouvoir y rajouter les algorithmes que l'on aura développé lors de la deuxième partie, et qui seront testés et intégrés lors de la troisième.

2.1 L'API Weka/Sources avec Eclipse

Pour explorer l'API, nous nous sommes aidés de l'IDE Eclipse, qui permet facilement parcourir les sources d'une librairie externe. Après avoir étudié l'arborescence des classes de l'API, nous avons pu cibler les différentes classes et méthodes qui nous intéressent, et étudié leurs fonctionnement. Nous nous sommes aidé de ce wiki [2].

2.2 L'utilisation des classes

Une fois familiarisés avec l'API Weka, on a creusé un peu plus du côté des classes qui pourraient nous être utiles pour ce TER. Il s'agit des certaines classes présentes dans le package "weka.classifiers". En effet, notre but étant d'intégrer des algorithmes de classification, il est utile de savoir comment tournent les algorithmes de classifications, leur paramétrage et l'architecture pour organiser les ressources pour ces derniers.
Quelques tests ont été menés notamment pour bayes naif multinomial, que nous avons fait tourné sur différentes données, et avec différentes options.

2.3 Ajout d'un algorithme dans Weka

Après avoir étudié en détail la classe *NaiveBayesMultinomial*, nous avons remarqué que le calcul des pondérations (dans l'implémentation de Weka, seul la mesure intra-classe Tf est utilisée) se fait dans la méthode **buildClassifier**. Nous allons donc créer une sous classe de *NaiveBayesMultinomial*, contenant une méthode surchargeant **buildClassifier** dans laquelle nous calculerons toutes les pondérations supplémentaires.

1. Weka est une suite populaire de logiciels d'apprentissage automatique. Écrite en Java, développée à l'université de Waikato, Nouvelle-Zélande. Weka est un Logiciel libre disponible sous la Licence publique générale GNU.
2. http://weka.wikispaces.com/

Une fois tout cela creusé et vu en détails, il faudra intégrer l'algorithme dans l'écosystème de Weka, c'est à dire, pour le rendre disponible dans l'Explorateur, expérimentateur, etc . Weka prend en charge les classes dérivées dans le package, ceci est géré par le *GenericPropertiesCreator*. Il faudra donc dire à Weka où trouver notre nouveau classificateur et il s'occupera de l'afficher dans la *GenericObjectEditor*.
Nous y reviendrons plus en détails lors de la troisième étape de notre TER : L'intégration des algorithmes dans WEKA.

3 Des nouvelles méthodes de classification

3.1 Pondérations intra-classes

//TODO

3.2 Pondérations inter-classes

//TODO

3.3 Algorithmes de classifications

//TODO

4 Développement des différentes classes

4.1 Méthodologie

//TODO

4.2 Extension de Naive Bayes Multinomial

//TODO

4.3 Class-Feature-Centroide

//TODO

5 Intégration et tests

5.1 Intégration dans l'écosystème de Weka

//TODO

5.2 Tests

//TODO

5.3 Résultats

//TODO

6 Discussion et Conclusion

//TODO

7 Sources

//TODO