Projet de Fin d'Études

- Keyword Extraction -

Rapport intermédiaire de PFE du Département Télécommunications, Services & Usages de l'INSA de Lyon

Batisse Alexandre
alexandre.batisse@insa-lyon.fr
Réalisation

Leaute Nathanaël
nathanael.leaute@insa-lyon.fr
Réalisation

Duffner Stefan
stefan.duffner@insa-lyon.fr
Encadrement

Garcia Christophe
christophe.garcia@insa-lyon.fr
Encadrement

30 novembre 2016

I Spécification détaillée du projet

La compréhension du langage naturel est l'un des principaux moteurs de la recherche sur l'intelligence artificielle. Ce domaine est en effet une évolution logique de l'abstraction toujours plus importante des langages informatiques, mais pas seulement. La compréhension du langage naturel reste un aspect fondamental de la pensée humaine et donc un passionnant sujet d'étude. Elle comprend notamment l'extraction de mot-clés, qui est elle-même une thématique de recherche très importante. Cette dernière est utilisée en premier lieu par les moteurs de recherche afin de retourner des documents. Avec l'essor des réseaux sociaux tels que Facebook, Linkedin, Twitter et des plateformes de discussions du type Reddit, Quora, StackExchange, la problématique d'extraction de mot-clés touche de plus en plus de domaines d'applications.

> *I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web — the content, links, and transactions between people and computers. A "Semantic Web", which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The "intelligent agents" people have touted for ages will finally materialize.*
> Tim Berners-Lee, *Weaving the Web*

Notre PFE s'inscrit dans la grande thématique du Web sémantique, notre but étant de créer un modèle capable d'annoter automatiquement un document texte avec un ensemble de mots-clés.

I.1 Contexte général

Kaggle[1] est une plateforme web qui organise des compétitions dans le domaine de la *Data Science*. Les entreprises ont la possibilité de soumettre des problèmes en science des données aux utilisateurs de la plateforme. À l'issue de la compétition, les participants à l'origine des solutions offrant les meilleurs résultats ou performances peuvent se voir offrir des prix.

Nous avons choisi pour ce projet de travailler sur une compétition Kaggle, et ce pour plusieurs raisons. Tout d'abord pour des questions de disponibilité des ressources, le fait de choisir un challenge comme point de départ nous permet d'avoir accès aux jeux de données et à des métriques d'évaluation fournis par le soumissionnaire du problème, ce qui garantit d'avoir les ressources de départ parfaitement adaptées à notre sujet. Par ailleurs, le classement des solutions étant publique, nous pouvons au terme de notre projet situer notre niveau de compétence par rapport à des professionnels et amateurs éclairés du monde entier, ce qui nous intéresse fortement en ce qui concerne notre potentielle carrière dans ce domaine. Enfin, la grande flexibilité permise dans le traitement de l'information et le regroupement de connaissances techniques sur la science des données sur une même plateforme permet d'assurer un apprentissage plus efficace des outils et technologies nécessaires à la création d'une solution.

I.2 Détail du sujet

Nous avons pour ce projet choisi le sujet Kaggle *Keyword Extraction*[2]. Il s'agit d'un challenge de recrutement proposé par *Facebook* en 2013 qui vise à prédire les *tags* (mots-clefs, sujets concernés, etc.) de discussions questions/réponses des sites *Stack Exchange*[3] en ne disposant que du titre du sujet et du corps de la question. Le jeu de données provient de différents sites Stack

1. `http://www.kaggle.com`
2. `https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction`
3. `http://stackexchange.com/`

Exchange et mêle questions techniques ainsi que non techniques. L'objectif de cette compétition est de proposer une solution comprenant au mieux de quoi traite une question, dans une optique de classification des fils de discussion en thèmes par exemple, mais non limité à cela. Nous pouvons parfaitement imaginer qu'une telle solution pourrait constituer une brique élémentaire de toute interface textuelle de communication Homme-Machine.

Le jeu de données fourni comprend un jeu d'apprentissage (2.19 Go en .csv.zip), un jeu de test (725.10 Mo en .csv.zip) ainsi qu'un exemple de fichier de soumission non utile à l'analyse.

Le fichier *Train.csv* est composé de quatre champs :

Id Identifiant unique pour chaque question

Title Titre de la question

Body Corps de la question

Tags mots-clefs associés à la question

Le fichier *Test.csv* est similaire à l'exception du champ *Tags* qui est à prédire. Le challenge consiste à construire ce dernier et à envoyer le fichier ainsi formé sur la plateforme de validation de la compétition.

II État de l'art

II.1 Multilabel learning

Nous allons introduire dans un premier temps le problème d'apprentissage multi-label (ML) ainsi que les principales approches permettant de résoudre celui-ci. Le *multi-label learning* est un problème d'apprentissage supervisé où chaque instance est associée à un ensemble de labels. Dans notre cas, nous travaillons avec des documents textes. Il s'agit alors d'associer chaque document avec un ensemble de tags. Le multi-label learning peut être vu comme une généralisation de l'apprentissage *multiclass*. Le problème est d'autant plus compliqué que l'on ne connait pas le nombre de tags à l'avance.

Plus formellement, on définit $\mathcal{X}$ l'espace des instances et $\mathcal{Y} = \{l_1, l_2, ..., l_k\}$ l'espace des labels. Chaque instance $X \in \mathcal{X}$ est associée à un sous-ensemble de labels pertinents $Y \subset \mathcal{Y}$.
Un sous-ensemble de tags Y peut être représenté sous la forme d'un vecteur binaire de dimension k, tel que $Y = (y_1, y_2, ..., y_k) = \{0, 1\}^k$ où $y_i = 1$ si le $i$ème label est pertinent pour le *topic*, nul sinon.

Gibaja et al.[1] présentent trois approches au problème de multi-label learning : *label ranking*, *classification multi-label* et *multi-label ranking*. Une approche de type label ranking consiste à prédire un score pour chaque label. Pour une instance X, on dispose alors d'un espace de classement pour l'ensemble des labels $\mathcal{Y}$ en deux partitions, Y étant la partition des labels pertinents et $\bar{Y}$ son complément. Enfin, le multi label ranking est présenté comme une combinaison des deux précédentes méthodes. Dans cette méthode, on produit à la fois une bipartition (Y,$\bar{Y}$) ainsi qu'un classement des labels. L'algorithme vise à ordonner les labels de Y avec un rang plus élevé que ceux de $\bar{Y}$. Par ailleurs, dans cette dernière approche, l'algorithme apprend également une fonction de seuil afin de déterminer le nombre de labels pertinents.

II.2 Binary Relevance

L'algorithme le plus simple pour résoudre un problème d'apprentissage multi-label est sans doute la méthode *Binary Relevance* (BR). Cette méthode transforme le problème en un ensemble de classifications binaires. Dans le paradigme de Binary Relevance, on entraine un classificateur binaire pour chaque label. Au moment de faire une prédiction, l'instance est envoyée

dans chaque classificateur et leurs résultats sont concaténés en un unique vecteur binaire. Le fait de décomposer le problème en classifications binaires présente plusieurs avantages (Luaces et al.[2]). Tout d'abord, l'ensemble des classificateurs peut être entrainé en parallèle. De plus, cette méthode a une complexité linéaire en fonction du nombre de labels. Enfin, il est possible d'ajouter de nouveaux labels sans avoir à entrainer de nouveau l'ensemble des classificateurs. L'approche Binary Relevance présente cependant certains défauts en comparaison d'autres modèles tels que les *Classifier Chains* (Jesse Read et al.[3]). En effet, BR fait l'hypothèse que les labels sont indépendants, ce qui est incorrect dans de nombreux problèmes. La conséquence directe est le fait d'entrainer deux classificateurs différents pour deux labels fortement corrélés, ce qui peut être vu comme redondant et inefficace. Enfin, dans le cas d'un *dataset* déséquilibré, certains labels sont beaucoup plus présents que d'autres, or l'approche Binary Relevance va entrainer chaque classificateur avec le même nombre de paramètres ce qui peut être perçu comme une consommation inutile des ressources. Malgré ces désavantages, BR reste une méthode robuste et sert de *benchmark* dans la plupart des problèmes d'apprentissage multi-label. Dans la partie suivante, nous nous intéressons à une approche permettant de tirer parti des corrélations pouvant exister entre les différents labels.

II.3 Réseaux de neurones

Nam et al.[4] ont récemment proposé une approche utilisant un réseau de neurones (NN) pour la classification multi-label de texte intitulée *Backpropagation for Multi-Label Learning* (BP-MLL). Cette approche est originellement issue d'un problème de classification dans le domaine de la génomique (Zhang et al.[5]). BP-MLL se présente comme un réseau de neurones standard composé d'une couche cachée. Il produit en sortie un vecteur de dimension k contenant le rang de chaque label. Enfin une fonction de seuil est également apprise pour prédire le nombre correct de labels.
En utilisant l'algorithme d'optimisation *Adagrad* (Duchi et al.[6]) ainsi que la technique de *Regularisation Dropout* (Hinton et al.[7]), l'approche BP-MLL constitue l'état de l'art en matière de classification multi-label. Ils ont également démontré que le réseau neuronal en question pouvait être entrainé en utilisant la fonction de *Cross Entropy* habituelle :

$$j(\theta, x, y) = -\sum_i y_i \log(o_i) + (1 - y_i) \log(1 - o_i) \tag{1}$$

Les deux modèles présentés ci-dessus utilisent le plus souvent des représentations de types *Bag-of-Words* ou TF-IDF (*Term Frequency - Inverse Document Frequency*). Ces approches sont tout à fait viables pour résoudre bon nombre de problèmes, mais elles présentent cependant toutes le défaut majeur de ne pas prendre en compte l'ordre des mots dans le texte. Les récentes avancées en matière d'espace vectoriel sémantique ont ouvert la porte à de nouveaux modèles permettant de vectoriser les documents. En effet, les modèles tels que *Word2Vec* (Mikolov et al.[8]) ou *GloVe* (Pennington et al.[9]) permettent désormais de représenter un mot par un vecteur de taille fixe appelé *word vector*. Ainsi un document textuel peut être représenté comme une matrice ou bien une séquence de word vectors.

II.4 Réseaux de neurones convolutifs

Les modèles dits *deep-learning* utilisant des réseaux de neurones convolutifs (CNN) constituent l'état de l'art dans le domaine de la vision par ordinateur (Szegedy et al.[10]). Grâce aux word vectors, les phrases peuvent être représentées par une matrice $m \times n$ où n est le nombre de mots dans la phrase et m la dimension des word vectors. Il est alors possible de réaliser différentes

convolutions sur cette phrase, de la même manière que sur une image. Une architecture de CNN a été proposée par Misha Denil et al.[11] afin de représenter un document texte dans une matrice de faible dimension.

Plus récemment, Kim[12] démontre que l'on peut entrainer un CNN pour différents problèmes liés au traitement du language naturel. Son modèle se révèle simple et atteint de très bonnes performances dans des nombreuses tâches telles que l'analyse de sentiments ou encore la classification de questions. Les CNNs présentent néanmoins un défaut lorsque l'on veut travailler avec du texte. Un *Convnet* n'accepte que des vecteurs de taille fixe en entrée, ce qui oblige le plus souvent à implémenter un *padding* pour gérer des phrases de différentes tailles. De plus, le nombre d'opérations réalisées dans un CNN est fixe et égal au nombre de couche présentes. C'est pourquoi nous allons détailler un autre type de modèle dans la partie suivante.

II.5 Réseaux de neurones récurrents

Les réseaux de neurones récurrents (RNN) font l'objet de beaucoup de recherches, notamment dans le domaine de la traduction machine (Sutskever et al.[13]). Les RNN travaillent sur des séquences de vecteurs et tentent de modéliser en quelque sorte la manière dont un humain lit un texte. Socher et al.[14] présentent une étude des RNN pour la classification de sentiments. Ils étudient en détail la manière dont les mots sont récursivement composés entre eux pour finalement classifier la phrase comme positive ou négative. Wang et al.[15] utilisent un RNN combiné à un CNN pour traiter le problème de la classification multi-label pour les images. Bien que leur travaux concernent les images, ils expliquent comment le RNN peut servir à encoder une séquence de labels.

Bibliographie

[1] Eva Gibaja and Sebastian Ventura. *Multi-Label Learning : A Review of the State of the Art and Ongoing 1 Research.* Data Mining and Knowledge Discovery, 2014.

[2] Oscar Luaces, Jorge Díez, José Barranquero, Juan José del Coz and Antonio Bahamonde. *Binary relevance efficacy for multilabel classification.* Progress in Artificial Intelligence, 2012.

[3] Jesse Read, Bernhard Pfahringer, Geoff Holmes and Eibe Frank. *Classifier Chains for Multi-label Classification.* In Proceedings of European conference on Machine Learning and Knowledge Discovery in Databases, 2009.

[4] Jinseok Nam, Jungi Kim, Eneldo Loza Mencia, Iryna Gurevych, and Johannes Fürnkranz. *Large-scale Multi-label Text Classification — Revisiting Neural Networks.* 2014.

[5] Min-Ling Zhang and Zhi-Hua Zhou. *Multi-Label Neural Networks with Applications to Functional Genomics and Text Categorization.* IEEE Transactions on Knowledge and Data Engineering, 2005.

[6] John Duchi, Elad Hazan and Yoram Singer. *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.* JMLR, 2011.

[7] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. *Dropout : A Simple Way to Prevent Neural Networks from Overfitting.* JMLR, 2014.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean. *Distributed Representations of Words and Phrases and their Compositionality.* NIPS, 2013.

[9] Jeffrey Pennington, Richard Socher and Christopher D. Manning. *GloVe : Global Vectors for Word Representation.* Stanford, 2014.

[10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens and Zbigniew Wojna. *Rethinking the Inception Architecture for Computer Vision.* CVPR, 2015.

[11] Misha Denil, Alban Demiraj, Nal Kalchbrenner, Phil Blunsom and Nando de Freita. *Modelling, Visualising and Summarising Documents with a Single Convolutional Neural Network.* Oxford, 2014.

[12] Yoon Kim. *Convolutional Neural Networks for Sentence Classification.* New York University, 2014.

[13] Ilya Sutskever, Oriol Vinyals and Quoc V. Le. *Sequence to Sequence Learning with Neural Networks.* EMNLP, 2013.

[14] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng and Christopher Potts. *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank.* EMNLP, 2013.

[15] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang and Wei Xu. *CNN-RNN : A Unified Framework for Multi-label Image Classification.* CVPR, 2016.