



UNIVERSITÉ PARIS DESCARTES

MASTER 2 MLSD

Text Mining: Reconnaissance d'entités nommées avec Spacy

Author:

Joseph GESNOUIN
Yannis TANNIER

Supervisor:

Mr François ROLE

January 5, 2019

Contents

1	Introduction	2
2	Natural Language Processing	3
2.1	Word embedding	3
2.2	Architecture Spacy	6
2.3	Named Entities Recognition	7
3	Base de données	8
3.1	Base de données utilisées	9
3.1.1	Online Mendelian Inheritance in Man (OMIM)	9
3.1.2	Diseases	9
3.1.3	Gene Ontology (GO)	9
3.1.4	Entrez Gene	9
3.1.5	Base de données PUBMED	10
3.2	Aggregation des données	10
4	Training et résultats	11
4.1	Apprentissage du modèle	11
4.2	Évaluation du modèle	12
5	Webpage de test du modèle	14
6	Conclusion	15
7	Bibliographie	16
8	Annexe	16

1 Introduction

De nos jours, la majorité des contenus scientifiques sont formalisés dans un format textuel. De plus, ces dernières années, l'accessibilité des publications scientifiques via les moteurs de recherche s'est améliorée de manière exponentielle, de telle sorte qu'il est de plus en plus aisé d'avoir accès à des publications de qualité et en nombre.

On estime que le nombre d'articles traitant des associations gènes-maladies disponibles dans les bases de données publiques augmente à une vitesse moyenne de 10 000 papiers par an, ce qui correspond à environ un papier par heure. Il en résulte une nécessité de trouver des méthodes plus optimales et plus rapides pour récupérer et traiter les connaissances textuelles qu'il est possible d'extraire de ces bases de données.

Alors qu'auparavant ce travail était réalisé par l'homme, l'avènement du Big Data et le flux de données qui en découle font qu'il est, aujourd'hui, potentiellement impossible pour un humain de mettre en place un système de classement manuel de toutes ces publications dans un temps acceptable. C'est la raison pour laquelle les algorithmes de text-mining sont de plus en plus souvent utilisés pour automatiser certaines requêtes afin de tirer le plus de connaissances possible de ce flot impressionnant de données dont la communauté scientifique ne disposait pas il y a encore quelques années.

C'est dans ce contexte que s'inscrit ce projet de Text-Mining: reconnaître deux types d'entités bio-médicales: Genes et Disease au sein de textes, afin de faciliter la détection de textes portant sur un de ces deux sujets.

2 Natural Language Processing

2.1 Word embedding

La notion de word embedding est assez basique: prenons une phrase simple: *J'ai presque fini de lire le livre de Machine Learning.*

À présent, considérons la même phrase à laquelle nous avons supprimé un mot, par exemple *livre*. On obtient la phrase *J'ai presque fini de lire le . de Machine Learning.*

Simplifions encore cette phrase en ne prenant que les trois mots avant le *.* et les trois mots après: *de lire le . de Machine Learning.*

En considérant cette phrase tronquée, si l'on demande à quelqu'un de trouver ce à quoi correspond le *.*, la majorité des gens répondront: livre, article ou papier.

C'est de cette manière que le contexte d'un mot nous permet de prédire le mot que ce contexte entoure. C'est également de cette manière que la machine peut découvrir que les mots livre, papier, article ont un sens similaire: ils partagent un contexte semblable dans plusieurs textes.

La méthode n'est pas nouvelle, c'est d'ailleurs le linguiste John Firth en disant "*You shall know a word by the company it keeps*" qui aurait aiguillé Thomas Mikolov et ses collaborateurs chez Google pour créer word2Vec: une méthode se focalisant sur l'apprentissage d'une représentation des mots. Elle permet de représenter chaque mot d'un dictionnaire par un vecteur de nombres réels correspondant dans un espace de dimension raisonnable. Les prolongements de mots constituent notamment une méthode pour mitiger un problème récurrent en IA: la dimension. En effet, sans les prolongements de mots, les objets mathématiques utilisés pour représenter les mots ont typiquement un grand nombre de dimensions, tant et si bien que ces objets se retrouvent isolés et deviennent épars. La technique des word embeddings diminue ainsi le nombre de ces dimensions.

Word2Vec et plus généralement le word embedding facilitent notamment l'analyse sémantique des mots. Cette nouvelle représentation des mots a ceci de particulier que les mots apparaissant dans des contextes similaires possèdent des vecteurs correspondants qui sont colinéaires dans l'espace des sémantiques.

Chose étonnante, la représentation sémantique des mots se manipule de manière conforme à notre intuition de la sémantique. On peut retrouver beaucoup de régularités linguistiques simplement en effectuant des translation linéaires dans cet espace de représentation. Par exemple, grâce à word2Vec on obtient le fameux:

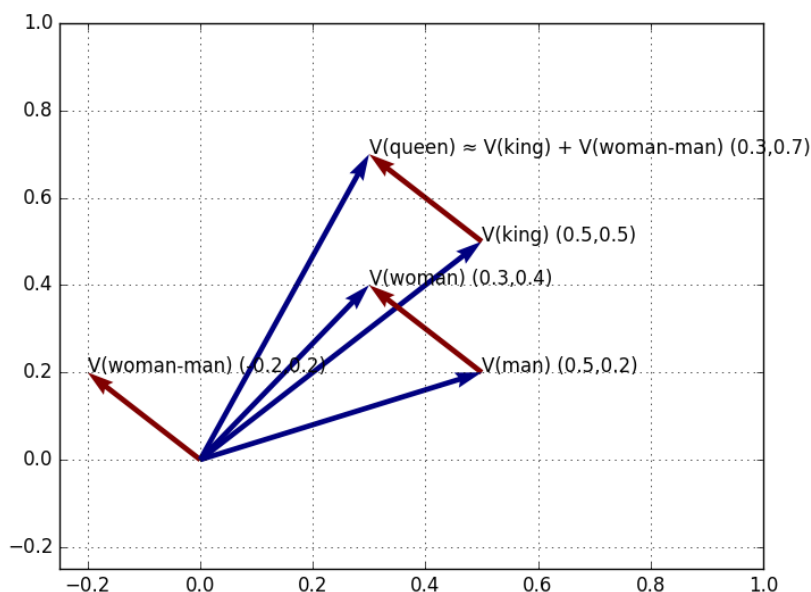


Figure 1: roi-homme+femme=reine représentation vectorielle

Ainsi, l'espace sémantique généré par les word2Vec semble s'être doté d'une arithmétique, et cette arithmétique semble être isomorphe à notre intuition de l'arithmétique du sens des mots.

Il existe deux approches au word2Vec:

- **Skip-gram (S-G)**: qui entraîne le réseau de neurones pour prédire un mot en fonction de son contexte, c'est à dire les mots avant/après dans une phrase: prédire les mots autour sur une fenêtre déterminée à l'avance à l'aide du mot étudié en entrée.
- **Continuous Bag of Words (CBOW)**: qui essaie de prédire le contexte en fonction du mot. L'entrée du réseau de neurones dans le cadre

du CBOW prend une fenêtre autour du mot et essaie de prédire le mot en sortie.

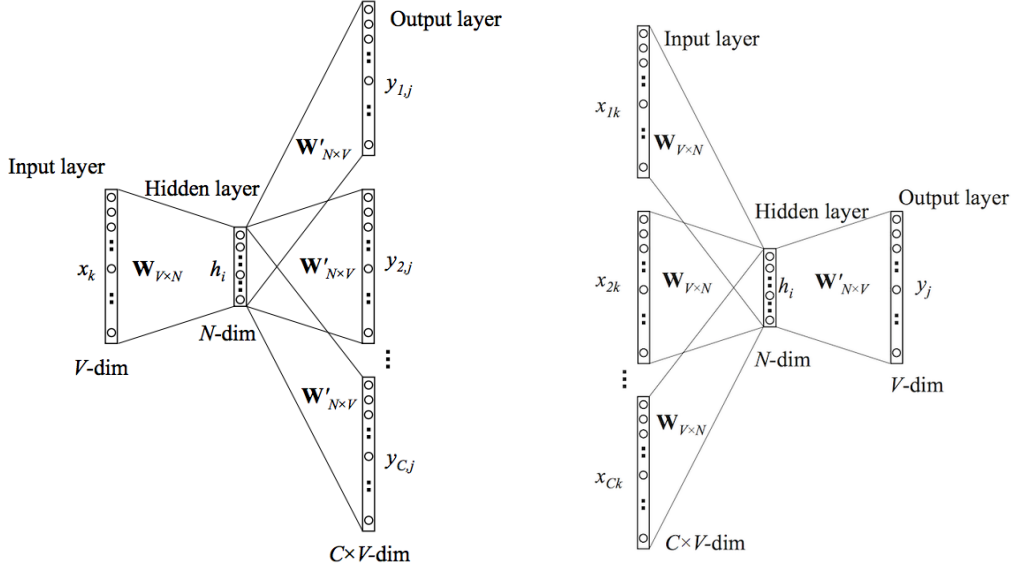


Figure 2: réseaux de neurones: S-G et CBOW

Le but de la fonction objectif de **Skip-gram (S-G)** est de maximiser la probabilité du mot en fonction de son contexte: en théorie de l'information, considérant un vecteur de mots W , il est possible d'estimer la probabilité de celui-ci:

$$\begin{aligned}
 P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2) \dots P(w_n|w_1^{n-1}) \\
 &= \prod_{k=1}^n P(w_k|w_1^{k-1})
 \end{aligned}$$

Cependant, le calcul des probabilités totales correspondant à la totalité de l'historique d'un mot dans un document peut très vite devenir complexe à calculer. L'intuition derrière le modèle n-gram réside dans l'approximation de l'historique des mots précédents grâce à seulement quelques mots et non la totalité de l'historique du vecteur: Ainsi, grâce à la propriété de Markov, il est possible d'approximer cette formule par:

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1})$$

Dans le cas du NER avec la librairie Spacy, nous utiliserons des modèles trigrammes pour reconnaître des genes ou des maladies, et nos mots seront caractérisés par des vecteurs de taille 180 dans l'espace des sémantiques.

2.2 Architecture Spacy

La majorité de nos traitements ont été réalisés via la librairie Spacy: Une librairie de traitement automatique du langage naturel focalisée sur la production et l'industrialisation de modèles.

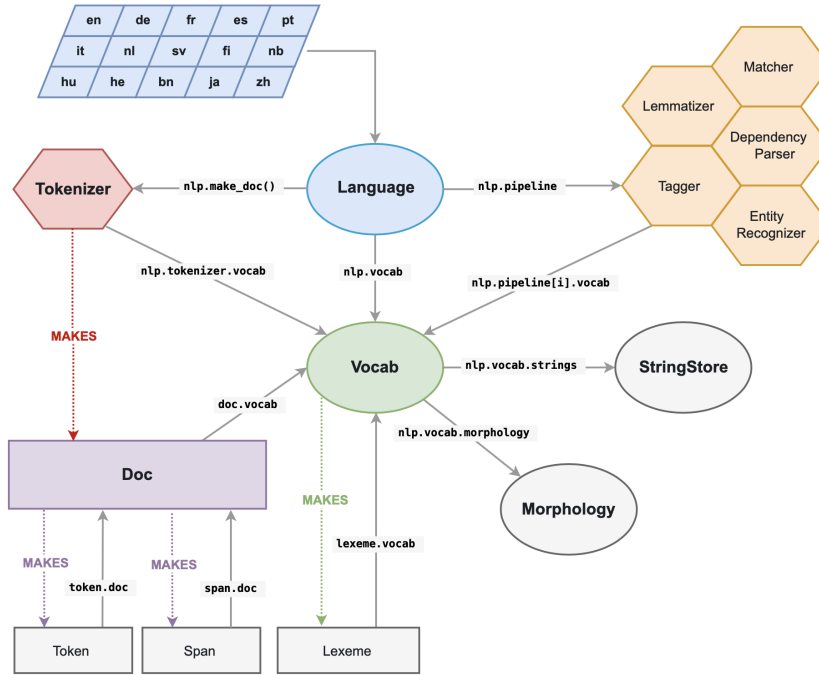


Figure 3: Architecture Spacy

Celle-ci étant complète et assez bien fournie, il nous a été possible de réaliser tout nos traitements: tokenization, embeddings, NER via cette librairie. Le code source du projet étant disponible [ici](#).

2.3 Named Entities Recognition

Une entité nommée est une "instance du monde réel" à laquelle on attribue un nom. Par exemple: une personne, un pays, un livre...

La librairie Spacy dispose d'un système de reconnaissance des entités extrêmement rapide qui attribue des labels à chacun des tokens. Les modèles par défaut proposés par Spacy indentifient une multitude d'entités nommées ou numériques telles que les entreprises, les locations géographiques, les organisations ou encore les produits.

Spacy propose également d'ajouter arbitrairement des classes à son système de reconnaissance d'entités, nécessitant de réentraîner le modèle avec des nouveaux exemples et c'est ce que nous avons réalisé.

Les modèles étant statistiques et dépendant fortement des exemples sur lesquels ils ont été entraînés, la NER, en règle générale ne marche pas parfaitement et nécessite assez souvent quelques réglages en fonction du cas d'usage.

Dans notre cas d'usage biomédical, nous avons rencontré plusieurs problèmes tel que des terminologies compliquées et irrégulières, des noms de maladies inédits, des noms différents représentant la même maladie, des structures syntaxiques compliquées référant à plusieurs maladies. Ces problèmes font partie des raisons majeures qui font que la détection automatique des noms de maladies ou de gènes est une tâche compliquée pour les algorithmes de NLP.

Dans l'état de l'art actuel, la plupart des algorithmes de DNER (Disease Named Entities Recognition) ont besoin d'utiliser manuellement des fonctionnalités plus complexes que pour certains autres cas d'usages (lexicaux, syntaxiques, sémantiques, morphologiques, dictionnaires, terminologie, embeddings). Ces fonctionnalités ajoutées de manière manuelle, requièrent des compétences spécifiques au domaine, linguistiques, mais requièrent également un temps supplémentaire à prendre en compte lors de la création du modèle.

Le fonctionnement d'un NER sous Spacy se divise en trois étapes distinctes:

- **Word embedding:** Utilisation de word2Vec afin de créer un vecteur de taille 180 caractérisant chacun des mots du dictionnaire. Cette caractérisation des mots ne prend pas en compte le contexte des mots.
- **Encode context:** Grâce à des réseaux de neurones type LSTM/CNN,

on transforme une multitude de vecteurs indépendants du contexte en une matrice portant de l'information sur celui-ci. Bien que la majorité des algorithmes de l'état de l'art actuel utilisent des réseaux LSTM, Spacy semble utiliser des réseaux de convolutions: ceux-ci étant plus simples à entraîner et à optimiser pour un résultat presque similaire.

- **Prediction:** grâce à un MLP basique, Spacy propose en output des labels ID, class ID, words...

3 Base de données

La croissance exponentielle des données notamment grâce aux phénomènes liés au Big Data est responsable de la grande diversité des bases de données. Les domaines d'application de ces bases de données sont donc assez diversifiés mais, lorsqu'elles s'appliquent à un même domaine, les bases de données tendent autant à se ressembler qu'à se compléter.

Dans le but d'être le plus précis possible, il aura fallu prendre en considération de nombreuses bases de données: en particulier, si l'on souhaite concevoir une base de données beaucoup plus grande. C'est ce que nous avons réalisé dans le cadre de ce projet.

Les bases de données utilisées sont nombreuses :

- Online Mendelian Inheritance in Man (OMIM)
- Diseases
- Gene Ontology (GO)
- Entrez-Gene (EG)
- PubMed

L'utilisation de toutes ces bases de données nous aura permis d'isoler une multitude de noms de maladies et de gènes grâce à des traitements de text-mining:

3.1 Base de données utilisées

3.1.1 Online Mendelian Inheritance in Man (OMIM)

La base de données OMIM rassemble de nombreux gènes humains ainsi que de nombreux phénotypes génétiques. Elle se concentre essentiellement sur la relation entre phénotype et génotype.

Dans le cadre de ce projet, seule une partie des données de cette base a été prise en compte. Il s'agit de la partie morbidmap , un résumé de la carte génétique humaine. Elle contient 6664 enregistrements d'associations gène maladie .

3.1.2 Diseases

La base de données DISEASES intègre des données probantes sur les associations entre les maladies et les gènes à partir de l'extraction automatique de texte, de la documentation manuelle, des données sur les mutations cancéreuses et des études d'association pangénomique.

Dans le cadre de ce projet, nous utiliserons la partie text mining full qui contient toutes les données de la base concernant les maladies de l'homme. Elle contient 641618 enregistrements d'associations gène maladie.

3.1.3 Gene Ontology (GO)

La base de données Gene Ontology provient d'un projet bio-informatique du même nom visant à étoffer et à structurer la description des gènes et des produits géniques dans le cadre d'une ontologie commune à toutes les espèces.

Dans le cadre de Gene Ontology, les gènes, les produits géniques et leurs propriétés sont décrits dans différentes bases de données.

Nous utiliserons, pour ce projet, les bases de données suivantes : la base de données Gene association for human references qui contient 354285 enregistrements et la base de données Gene ontology terms qui contient 37104 enregistrements.

3.1.4 Entrez Gene

ENTREZ-GENE est une base de données spécifique aux gènes créée par le NCBI. Celle-ci décrit et intègre des informations et propriétés diverses sur

les gènes et, en particulier, leur nomenclature, leurs descriptions sommaires, les accessions de séquences géniques et spécifiques au produit, la localisation chromosomique, les rapports de voies et interactions protéiques, les marqueurs associés et les phénotypes.

La base de données Entrez-Gene propose 19063 enregistrements.

3.1.5 Base de données PubMed

La base de données PubMed a été développée par le NCBI. Le moteur de recherche qui y est associé est gratuit et donne accès à la base de données bibliographique MEDLINE, rassemblant des citations et des résumés d'articles de recherche biomédicale.

3.2 Aggregation des données

Grâce à l'ensemble de ces bases de données et après avoir agrégé ces données en fonction de leur provenance, nous disposons d'une liste assez conséquente des maladies et gènes existants. Ainsi, nous disposons d'environ 4 000 maladies différentes et de 16 000 gènes.

De cette manière nous avons accès à un nombre non négligeable de maladies que nous pouvions directement chercher dans les fichiers asthma et autism fournis au début du projet afin d'annoter de manière automatique nos futures données d'apprentissage.

```
Cholera
Primary bacterial infectious disease
Bacterial infectious disease
Disease by infectious agent
Disease
Cerebral malaria
Plasmodium falciparum malaria
Kidney failure
Adult respiratory distress syndrome
Kidney disease
Urinary system disease
Lysosomal storage disease
Mucopolysaccharidosis
Inherited metabolic disorder
Lipid storage disease
Sphingolipidosis
Mucopolysaccharidosis VII
Disease of metabolism
Disease
Aspartylglucosaminuria
Neuronal ceroid lipofuscinosis
Mucopolipidosis
Fabry disease
Wolman disease
Mucopolysaccharidosis III
Niemann-Pick disease
Metachromatic leukodystrophy
non-langerhans-cell histiocytosis
Carbohydrate metabolic disorder
Glycogen storage disease II
alpha-mannosidosis
Fucosidosis
```

Figure 4: Extrait des maladies récupérées grâce à l'aggregation de ces bases de données

4 Training et résultats

4.1 Apprentissage du modèle

Une fois ces premiers traitements de données réalisés afin d’obtenir des noms de maladies et de gènes, nous avons considéré plusieurs approches pour entraîner notre modèle spacy: prendre 70% de chacun des deux fichiers fournis: asthma et autism, les annoter et tester le modèle sur les 30% restants ou bien utiliser la totalité d’un fichier comme données d’apprentissage et tester le modèle de NER sur le fichier restant.

De ces deux solutions envisagées nous avons préféré la seconde. La tokenization étant une des procédures standard de preprocessing, nous n’avons pas omis cette étape: le but étant de partager les phrases d’un fichier en unités atomiques. Tout simplement, nous avons tokenisé chaque phrase grâce à certains caractères spécifiques.

Ainsi, nous avons commencé par diviser le jeu de données asthma et en avons cherché les occurrences des maladies et gènes dont nous avons le nom grâce aux données précédemment agrégées. Ainsi, il nous a été aisé d’annoter de manière automatique le jeu de données asthma.

Une fois notre jeu de données d’apprentissage annoté, il nous suffisait d’utiliser la librairie Spacy afin d’entraîner notre modèle. Nous avons eu l’occasion de faire la totalité de nos apprentissages sur une RTX 2080 TI. De ce fait nous avons eu le temps, et l’avantage d’essayer une multitude de modèles différents, pré-entraînés ou non, en faisant varier les paramètres à outrance mais également en ne lésinant pas sur le nombre d’époques utilisés pour chaque modèle. Au final, afin d’optimiser au mieux notre modèle, nous avons suivi les recommandations de spacy qui semblent fonctionner correctement pour une majorité des problèmes de NLP.

- Utilisation de l’Adam solver pour fixer le pas d’apprentissage.
- Utilisation de la L2 régularisation: permettant de contrecarrer cet effet d’explosion des gradients en vérifiant la taille des poids du réseau et en appliquant une pénalité en fonction de la valeur de la taille des poids.
- Utilisation d’un dropout rate élevé avec une déclinaison linéaire au fil de l’apprentissage afin d’éviter au modèle de sur-apprendre dès le début de l’apprentissage, le jeu d’apprentissage étant de taille limitée.

- Utilisation des poids moyens du réseau lors de l'apprentissage et non des dernières valeurs d'apprentissage, c'est une technique popularisée par Collins qui semble être utilisée assez souvent pour des problèmes de NLP: en pratique il s'agit juste de garder en mémoire le moving average des poids durant la totalité du training.

4.2 Évaluation du modèle

En utilisant des indicateurs classiques d'évaluation de système de NER, il nous est possible d'avoir un retour de cohérence sur notre modèle:

$$\begin{aligned} precision &= \frac{TP}{TP + FP} \\ recall &= \frac{TP}{TP + FN} \\ F1 &= \frac{2 \times precision \times recall}{precision + recall} \end{aligned}$$

Nous avons décidé d'évaluer notre modèle de deux manières différentes, la première, comme premier indicateur de cohérence, était d'évaluer notre modèle sur le fichier n'ayant pas servi à l'apprentissage: autism, sur lequel nous obtenons les résultats suivants:

- **Precision:** 90.7%
- **Recall:** 85.4%
- **F:** 87.9%

Ces résultats nous permettant de situer notre modèle sur un des deux fichiers fournis. Les résultats étant assez encourageants mais ne nous permettant pas de comparer la solidité de notre modèle vis-à-vis des autres méthodes de NLP actuelles car l'évaluation se faisant sur un fichier peu connu. Nous avons souhaité approfondir l'évaluation de notre modèle de reconnaissance de données aux autres méthodes de NER en testant notre modèle sur le jeu de donnée **NCBI disease corpus**.

NCBI est un jeu de données servant de ressource à la communauté biomédicale traitant des sujets de NLP et de NER. Celui-ci est composé de 793 abstracts

provenant de papiers PubMed et mentionne plus de 6800 maladies.
À ce jour, l'état de l'art sur NCBI est d'environ 85%:

Method	P	R	F
BANNER	83.80	80.00	81.80
Bi-LSTM + WE	84.87	74.11	79.13
MCNN	85.08	85.26	85.17

Sur ce jeu de données, nos résultats avec notre modèle Spacy sont de l'ordre de:

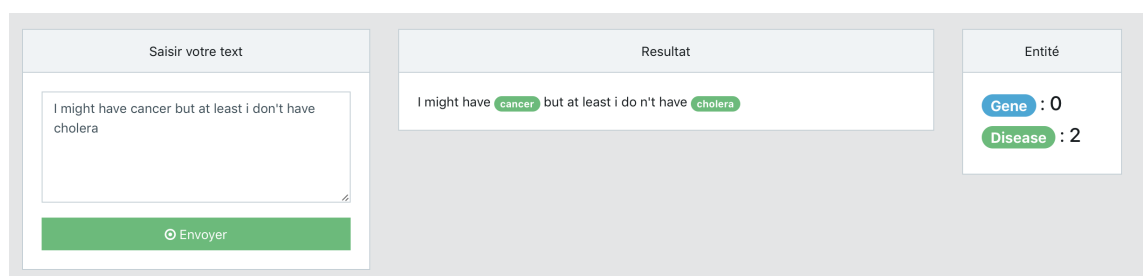
- **Precision:** 85.2%
- **Recall:** 88.3%
- **F:** 82.3%

Le problème étant que notre modèle étant entraîné à reconnaître les gènes et les maladies, nos résultats sont probablement "gonflés" par la détection des gènes et non seulement des maladies pour ce jeu de données. Ainsi il nous est impossible de savoir exactement comment celui-ci se situe vis à vis de l'état de l'art, même si une chose est assurée: il semble assez représentatif de ce qu'il est possible de réaliser avec un modèle de NER sur des problèmes complexes tels que la reconnaissance d'entités nommées bio-médicales. Ainsi notre modèle semble assez robuste si on le compare à d'autres méthodes de NLP de l'état de l'art sur NCBI.

5 Webpage de test du modèle

Afin de pouvoir approfondir le sujet et potentiellement améliorer le modèle, nous avons également mis en place un système de test personnalisé de notre modèle disponible **ici** ou en annexe.

Celui-ci permettant à un utilisateur de tester n'importe quel texte qu'il jugerait bon d'essayer sur le modèle afin de récupérer les résultats pour ses documents:



The screenshot displays a web interface for testing a model. It is divided into three main sections:

- Saisir votre text**: A text input area containing the sentence "I might have cancer but at least i don't have cholera". Below the input is a green button labeled "Envoyer".
- Resultat**: A section showing the processed text: "I might have **cancer** but at least i do n't have **cholera**". The words "cancer" and "cholera" are highlighted in green boxes.
- Entité**: A section showing the results of the model's classification: "Gene : 0" and "Disease : 2". The word "Disease" is highlighted in a green box.

Figure 5: Exemple d'utilisation de notre webpage pour tester le modèle spacy

Le but étant de découvrir des points d'amélioration pour rendre meilleure la qualité du modèle, et, à terme, envisager l'implémentation d'une méthode de progressive learning du modèle sur ce même site.

Nous avons remarqué l'existence de Prodigy, un outil d'annotation de NLP et nous aurions aimé pouvoir finaliser ce projet en ajoutant un accès à prodigy sur notre webpage afin qu'un utilisateur puisse contribuer à l'amélioration de notre modèle par retour de cohérence grâce au progressive learning. Malheureusement, nous n'avons à ce jour toujours pas eu de retour de leur équipe afin de nous accorder une licence et c'est pour cela que notre propre webpage de test a vu le jour.

6 Conclusion

Durant ce projet, nous avons eu l'occasion de nous familiariser avec certains outils et algorithmes utilisés fréquemment en text-mining tels que word2Vec ou plus généralement les word embedding. Nous avons eu également l'occasion de prendre en main l'outil Spacy: une librairie spécialisée dans le traitement du langage à orientation industrielle et ainsi continuer notre pratique des réseaux de neurones et des spécificités à prendre en compte pour optimiser un réseau de neurones à orientation NLP.

Il nous a été possible de monter de toute pièce un modèle de detection automatique d'entités bio-médicales suffisamment robuste en se fiant à notre évaluation de celui-ci. La nature de ce projet nous a permis de traiter une problématique de bout en bout: de la selection des bases de données en passant par leur aggregation jusqu'à la mise en production du modèle final qui se concrétise par notre webpage où l'utilisateur peut directement tester le modèle construit.

Ce projet a donc été formateur de par son orientation industrielle: la majorité du travail réalisé lors de celui-ci étant un travail d'aggregation des données afin d'avoir un ensemble d'apprentissage cohérent grâce à l'utilisation de plusieurs bases de données bio-médicales connues.

Ces étapes préalables à l'entraînement d'un modèle sont trop souvent déjà réalisées de nos jours: des centaines de jeux d'entraînement prêts à l'utilisation sont disponibles sur le net et de ce fait, avoir eu l'occasion de traiter le sujet du début à la fin nous aura permis de mieux comprendre l'importance et l'impact sur le résultat final de disposer d'un jeu de données bien construit.

7 Bibliographie

References

- [1] William B. Cavnar and John M. Trenkle *N-Gram-Based Text Categorization*.
- [2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean *Distributed Representations of Words and Phrases and their Compositionality*.
- [3] Scharolta Katharina Siencnik *Adapting word2vec to Named Entity Recognition*.
- [4] Emma Strubell, Patrick Verga, David Belanger, Andrew McCallum *Fast and Accurate Entity Recognition with Iterated Dilated Convolutions*.
- [5] Ridong Jiang, Rafael E. Banchs, Haizhou Li *Evaluating and Combining Named Entity Recognition Systems*.
- [6] Michael Collins, Brian Roark *Incremental Parsing with the Perceptron Algorithm*.

8 Annexe

Code source: <https://github.com/yannistannier/textmining>

Home pour tester le modèle: <http://textmining.yannistannier.io>