Participation d'EDF R&D à DEFT 2020

Danrun Cao, Alexandra Benamar, Manel Boumghar, Meryl Bothua, Lydia Ould Ouali, Philippe Suignard

Participation d'EDF R&D à DEFT 2020

Danrun Cao[1], Alexandra Benamar, Manel Boumghar, Meryl Bothua, Lydia Ould-Ouali, Philippe Suignard
EDF R&D, 7 boulevard Gaspard Monge, 91120 Palaiseau
prenom.nom@edf.fr

RESUME

Ce papier décrit la participation d'EDF R&D à la campagne d'évaluation DEFT 2020. Notre équipe a participé aux trois tâchés proposées : deux tâches sur le calcul de similarité sémantique entre phrases et une tâche sur l'extraction d'information fine autour d'une douzaine de catégories. Aucune donnée supplémentaire, autre que les données d'apprentissage, n'a été utilisée. Notre équipe obtient des scores au-dessus de la moyenne pour les tâches 1 et 2 et se classe 2e sur la tâche 1. Les méthodes proposées sont facilement transposables à d'autres cas d'application de détection de similarité qui peuvent concerner plusieurs entités du groupe EDF. Notre participation à la tâche 3 nous a permis de tester les avantages et limites de l'outil SpaCy sur l'extraction d'information.

ABSTRACT

This paper describes the participation of EDF R&D at DEFT 2020 evaluation campaign. Our team participated in the three proposed tasks: two of them on Semantic Similarity Detection and one on Information Extraction in Clinical Cases. No additional data other than the training data was used. Our team gets above average results for the first and the second task and got the second place on the second task. The proposed methods are easily transferable to other Semantic Similarity Detection use cases and may interest several entities of the EDF group.

MOTS-CLES : données cliniques, détection de similarité sémantique, Word2Vec, graphes sémantiques, extraction d'information.

KEYWORDS: clinical data, Semantic Similarity Detection, Information Extraction, Word2Vec, semantic graphs.

1 Introduction

Dans la continuité de DEFT 2019, l'édition 2020 du défi fouille de textes (Cardon, 2020) continue d'explorer les cas cliniques rédigés en français. Cette nouvelle édition porte sur l'extraction

[1] Contribution égale de tous auteurs, listés par ordre alphabétique

d'information fine autour d'une douzaine de catégories. En dehors du domaine clinique, deux tâches sont proposées sur le calcul de similarité sémantique entre phrases. Plusieurs éléments nous ont motivés à participer à cette édition du défi. Participer à DEFT est l'occasion :

- de tester plusieurs méthodes de calcul de similarité dont les résultats contribuent directement à EDF Commerce et à d'autres entités du groupe EDF (tâches 1 et 2).

- d'évaluer des outils d'extraction d'entités nommées, comme SpaCy, même si ce travail est réalisé sur des données différentes des données EDF, comme ici avec des données médicales (tâche 3).

2 Description des tâches et méthodes utilisées

2.1 Tâche 1 : « Identifier le degré de similarité entre paires de phrases parallèles et non-parallèles sur plusieurs domaines »

2.1.1 Présentation

La tâche 1 consiste à déterminer le niveau de similarité entre paires de phrases, sur une échelle allant de 0 à 5 :

- 5 : Les deux phrases sont complètement équivalentes, car elles veulent dire la même chose ;
- 4 : Les deux phrases sont pour la plupart équivalentes, mais certains détails sans importance diffèrent ;
- 3 : Les deux phrases sont à peu près équivalentes, mais certaines informations importantes diffèrent ou manquent ;
- 2 : Les deux phrases ne sont pas équivalentes, mais partagent quelques détails ;
- 1 : Les deux phrases ne sont pas équivalentes, mais portent sur le même sujet ;
- 0 : Les deux phrases sont complètement différentes.

Exemple, avec une similarité de 4 entre les deux phrases suivantes :

- En l'absence d'amélioration comme en cas de persistance des symptômes, prendre un avis médical.
- En l'absence d'amélioration comme en cas de persistance des symptômes au-delà de 7 jours de traitement, prenez un avis médical.

La difficulté de la tâche consiste à affecter une note de manière absolue. Pour résoudre cette difficulté, les méthodes que nous proposons reposent toutes sur un apprentissage, puisqu'un jeu de données de 600 paires de phrases était fourni et annoté. Nos méthodes consistent à extraire des *features* pour ces différentes phrases, à calculer des similarités ou distances entre les paires de phrases, puis à entraîner un classifieur.

2.1.2 Run 1 : Dice + RL

Les différentes étapes sont les suivantes, pour une paire de phrases donnée :

- Suppression de la ponctuation et des mots faisant partie d'une stop-liste ;
- Troncature des mots à 6 caractères : « biberon » et « biberons » sont transformés en « bibero » ;
- Pour chaque phrase, ne sont gardés que les mots uniques ;
- Calcul des « features » ou descripteurs suivants :

1. Sorensen-Dice (Dice, 1945) : 2 fois le nombre de mots en commun divisé par la somme des nombres de mots de chaque phrase.

$$sim = \frac{2 * |X \cap Y|}{|X| + |Y|}$$

2. Le nombre de mots en commun entre les deux phrases
3. Le nombre de mots de la phrase 1
4. Le nombre de mots de la phrase 2
5. L'écart = nombre de mots de la phrase 1 - nombre de mots de la phrase 2
6. La valeur absolue de l'écart précédent
7. La distance de Levenshtein entre les deux phrases

Les descripteurs ont été calculés pour chaque paire du corpus d'apprentissage puis convertis au format ARFF pour être utilisées au sein du logiciel WEKA (Hall, 2009). Deux classifieurs ont été testés : « Régression Logistique » et « Random Forest ». La régression logistique obtenait les meilleurs résultats sur le corpus d'apprentissage, c'est donc cette méthode qui a été utilisée pour la phase de test.

2.1.3 Run 2 : Graphes sémantiques + RL

Dans cette méthode, nous nous appuyons sur des graphes sémantiques obtenus à partir d'arbres syntaxiques dont nous éliminons les informations jugées non pertinentes. Ensuite pour chaque paire de phrases, nous alignons les graphes sémantiques et extrayons des *features*. Ces *features* constituent notre entrée pour l'entraînement d'un classifieur, en l'occurrence une Régression Logistique.

2.1.3.1 Graphes sémantiques

L'analyse syntaxique est effectuée avec le schéma d'annotation UD (Nivre et al.,2016) du logiciel Talismane (Urieli, 2013). Les arbres syntaxiques sont décrits sous le format CoNLL. Ces arbres sont ensuite simplifiés afin d'éliminer les éléments grammaticaux. À l'issu de cette étape de simplification, nous obtenons un graphe sémantique. Les critères de simplifications des arbres syntaxiques sont :

- Suppression de la ponctuation.
- Suppression des mots présents dans une « stop-liste ».
- Suppression de certaines catégories de mots : *DET* (déterminant), *ADP* (préposition), *PRON* (pronom), *CCONJ* (conjonction de coordination), *ADP+DET* (déterminant composé), *PART* et *X* (particule et partie de locution qui ne correspond à aucune catégorie traditionnelle).
- Suppression des nœuds ayant comme dépendance : *det* (déterminant), *mark* (mot introduisant une proposition de subordination), *punct* (ponctuation), *cc* (marqueur de

conjonction de coordination), *case* (préposition), *fixed* (structuré figée non grammaticale), *reparandum* (reformulation), *discourse* (mot de discours), *cop* (verbe copule), *aux* (verbe auxiliaire).

Si les nœuds supprimés ont des descendants, ces derniers sont rattachés à la racine du nœud supprimé.

2.1.3.2 Alignement de graphes

L'alignement des graphes se fait au niveau des nœuds afin de trouver les termes similaires entre les phrases sources et les phrases cibles : nous produisons une matrice de distance d'édition entre les mots au sein de deux phrases et nous appliquons l'algorithme hongrois pour trouver le meilleur alignement entre les arbres syntaxiques simplifiés. La distance d'édition entre les mots est définie par leur distance cosinus, calculée avec un modèle word2vec pré-entraîné par Jean-Philippe Fauconnier[2]. En cas de mots hors vocabulaire, nous comparons les lemmes : s'il s'agit du même lemme, la distance minimum (=0) est attribuée, sinon la distance maximum (=2). Comme les deux phrases n'ont souvent pas la même longueur, nous insérons des nœuds vides pour assurer que chaque mot ait un match. Tout match avec un nœud vide reçoit la distance maximum (=2). Concernant les types de match entre nœuds, il en existe trois : le *match exact* où il s'agit du même mot, le *match loose* où deux mots différents sont alignés et le *match dummy* où l'un des nœuds alignés est un nœud vide.

Les arêtes sont représentées par les nœuds qu'elles relient. Pour chaque arête source, nous cherchons les nœuds cibles qui correspondent aux nœuds sources en question. Le match de l'arête source est donc le chemin le plus court qui relie les nœuds cibles trouvés. Ce match ne correspond donc pas forcément à une arête réelle dans le graphe, il peut en nécessiter plusieurs. La distance d'édition de ce match est calculée ainsi :

$$DE[(a1, b1) \rightarrow (a2, b2)] = (|DE[a1 \rightarrow b1]| + |DE[a2 \rightarrow b2]|) \times NbArêtes$$

Il existe également trois types de match entre arêtes : le *match exact* où une seule arête cible est nécessaire, le *match loose* où plusieurs arêtes cibles sont nécessaires et le *match dummy* où l'un des nœuds cibles est un nœud vide.

2.1.3.3 Extraction de features et entraînement

Nous avons défini trois grandes familles de *features* ci-dessous que nous avons extraites :

1. Au niveau de phrase entière :
   - Différence de nombre de mots,
   - Distance Damerau-Levenshtein de deux phrases au niveau du caractère,
   - Distance Damerau-Levenshtein de deux phrases au niveau du mot.

2. Au niveau de mot/nœud de graphe :
   - Distance d'édition (DE) absolue : la somme d'DE des matches de nœuds
   - Distance d'édition relative : DE absolue/DE maximum (toutes les DE à maximum)

[2] http://fauconnier.github.io/

- Nombre et pourcentage de *matches exact*
- Nombre et pourcentage de *matches loose*
- Nombre et pourcentage de *matches dummy*

3. Au niveau dépendance/arête de graphes :
   - Différence entre le nombre d'arêtes
   - Nombre et pourcentage de *matches exact*
   - Nombre et pourcentage de *matches loose*
   - Nombre et pourcentage de *matches dummy*
   - Distance d'édition absolue : la somme d'DE des matches d'arêtes
   - Distance d'édition relative : DE absolue/DE maximum (toutes les DE à maximum)

Nous obtenons donc un total de 20 *features*. Plusieurs classifieurs ont été testés également : Régression Logistique et Random forest. Si les deux classifieurs renvoient des résultats similaires, la régression logistique est plus stable. Nous avons donc retenu cette dernière.

2.1.4 Run 3 : Features + RL

Les différentes étapes pour ce run pour une paire de phrase donnée, sont les suivantes :

- Suppression de la ponctuation, des espaces multiples et des mots faisant partie d'une stop-liste.
- Calcul des *features* ou descripteurs suivants (Cardon, 2018) :

1. Le nombre de mots en commun entre les deux phrases ;
2. Le pourcentage de mots de la phrase 1 inclus dans la phrase 2 ;
3. La différence de la longueur des deux phrases ;
4. La différence de la longueur moyenne des mots entre les deux phrases ;
5. Le nombre des *n-grams* en commun entre les deux phrases. Nous avons calculé les *bi-grams*, *tri-grams* et *n-grams* de longueur 4 ;
6. La distance de Levenshtein entre les deux phrases ;
7. La distance de Jaccard entre les deux phrases ;
8. La distance sémantique (Word Mover Distance) entre les deux phrases.

Les descripteurs ont été calculés pour chaque paire de phrases au niveau du mot sauf la distance de Levenshtein qui a été calculée au niveau des caractères et au niveau des mots.

Deux classifieurs ont été testés à savoir « Régression Logistique » et « Random Forest ». La régression logistique obtenait les meilleurs résultats sur le corpus d'apprentissage car plus stable, c'est donc cette méthode qui a été retenue pour la phase de test.

2.1.5 Résultats

Méthode	Evaluation en EDRM
Maximum	*0,8217*
Run 1 : Dice + RL	***0,8198***
Run 3 : Features + RL	***0,8069***
Run 2 : Graphes sémantiques + RL	***0,8018***
Médiane	*0,7947*
Moyenne	*0,7617*
Minimum	*0,6533*

Table 1 : résultats de la tâche 1

Les résultats de nos trois *run* se trouvent tous au-dessus de la moyenne. Notre premier *run* est très proche du meilleur résultat obtenu (0,0019 d'écart). Nous avons observé dans les sorties que les différences entre les valeurs 0 et 1 ainsi qu'entre les valeurs 4 et 5 sont parfois difficiles à appréhender, même pour un humain. En voici un exemple :

- « Boris Godounov meurt, subitement, le 13 avril 1605 à Moscou : on parla alors d'empoisonnement ou de suicide. »
- « Boris Godounov est un monarque russe qui régna de 1598 jusqu'à sa mort en 1605 sur la Russie. »

La note à prévoir ici était 0 (note pour deux phrases complètement différentes), alors qu'elles parlent de Boris Godounov. La note aurait pu être 1 (note pour deux phrases ne sont pas équivalentes, mais qui portent sur le même sujet), le sujet étant la mort de B. Godounov.

2.2 Tâche 2 : « identifier les phrases parallèles possible pour une phrase source »

2.2.1 Présentation

La tâche 2 est une tâche d'appariement entre une phrase source et trois phrases cibles. L'objectif consiste à trouver quelle phrase, parmi les trois phrases cibles, est la plus proche de la phrase source. Exemple : la phrase la plus proche de la phrase source est la phrase cible n°2 :

- Source : « compte tenu des données disponibles, l'utilisation chez la femme enceinte ou qui allaite est possible ponctuellement » ;
- Cible 1 : « ce médicament est un laxatif utilisé par voie orale » ;
- Cible 2 : « ce médicament, dans les conditions normales d'utilisation, peut être utilisé ponctuellement pendant la grossesse et l'allaitement » ;
- Cible 3 : « boîte de 1 flacon de 250 ml ou 500 ml ».

Nous proposons deux méthodes différentes, l'une basée sur un calcul de similarité de type Dice et l'autre sur des embeddings de documents.

2.2.2 Run 1 : Dice

La première méthode est similaire à la méthode 1 de la tâche 1. Elle consiste à éliminer les mots faisant partie d'une stop-liste, à garder les mots uniques puis à calculer le coefficient dice-sorensen entre la phrase source et les trois phrases cibles, puis à choisir la phrase qui maximise cette similarité.

2.2.3 Run 2 : USE

Cette méthode est basée sur une représentation vectorielle des phrases calculée à l'aide de USE (Universal Sentence Encoder) (CER *et al.*, 2018), puis à calculer une similarité cosinus entre la phrase source et les 3 phrases cibles puis à choisir la phrase qui maximise cette similarité. Pour cette tâche, aucun prétraitement n'a été effectué. Le modèle encodeur de USE utilisé est « Universal Sentence Encoder Multilingual module », proposé par Google en 2019, appris sur 16 langues dont le français.

2.2.4 Résultats

Méthode	Evaluation en MAP	Nb d'erreurs
Maximum	*0,9906*	*5*
Médiane	*0,9868*	
Run 2 : USE	**0,9867**	**7**
Run 1 : Dice + RL	**0,9830**	**9**
Moyenne	*0,9822*	
Minimum	*0,9396*	*32*

Table 2 : résultats de la tâche 2

Les résultats de nos deux *run* se trouvent au-dessus de la moyenne. Notre deuxième *run* est très proche du meilleur résultat obtenu (0,0039d'écart) ainsi que le premier (0,0076). Le gagnant a commis 5 erreurs, notre *run 2* 7 et notre *run 1* 9 erreurs. Ces différences sont facilement explicables. Par exemple, la méthode du *run 1* se trompe sur la phrase « les stéroïdes oraux sont le traitement standard » en lui affectant la cible « les effets indésirables les plus courants associés aux stéroïdes oraux sont la prise de poids et l'augmentation de la pression artérielle » au lieu de « les corticostéroïdes par voie orale sont le traitement le plus courant ». Cela s'explique par les différences entre les mots employés : « corticostéroïde »s au lieu de « stéroïdes », voie « orale » au lieu de « oraux » et « courant » au lieu de « standard ». La méthode du *run 1* se trompe s'il y a plusieurs synonymes ou expressions légèrement différentes entre la source et la bonne cible.

2.3 Tâche 3 : « extraction d'information »

La tâche 3 est une tâche d'extraction d'information. Elle consiste à repérer, dans les cas cliniques, les informations fines autour d'une dizaine de catégories. Quatre domaines sont couverts :

- autour des patients : anatomies ;
- autour de la pratique clinique : examen, pathologie, signe ou symptôme ;

- autour des traitements médicamenteux et chirurgicaux : substance, dose, durée, fréquence, mode d'administration, traitement (chirurgical ou médical), valeur ;
- autour du temps : date, moment.

2.3.1 Run 1 : Spacy

Nous avons adopté des approches symboliques pour extraire les informations recherchées. Nous avons ainsi utilisé la bibliothèque python SpaCy[3], et plus précisément ses composants de reconnaissance d'entités nommées personnalisées. Bien que l'outil propose de nombreuses fonctionnalités intéressantes, une contrainte importante est que chaque *token* ne peut correspondre qu'à un seul type d'entité nommée. Ceci a été problématique pour les syntagmes complexes, notamment pour les catégories « pathologie » et « sosy ». Nous avons tout de même essayé de repérer des « pathologies », mais aucune prédiction n'a été faite pour « sosy ». Concernant les catégories restantes, nous décrivons ci-dessous les règles développées.

2.3.1.1 Règles lexicales

Il s'agit des règles basées sur des lexiques spécifiques. Ces règles permettent de récupérer les termes n'ayant pas ou peu de variations. Nous avons utilisé ce type de règles pour les catégories « substance », « état », « prise », « changement », « norme », « assertion ». Les lexiques sont la plupart extraits du corpus d'entraînement, à l'exception de « substance » pour lequel nous avons introduit une liste de spécialités pharmaceutiques et leurs compositions provenant du site officiel de l'ANSM[4].

2.3.1.2 Règles à la base des patterns morphologiques

Certaines catégories, telles que « date », « dose », « mode » et « fréquence », se caractérisent par un nombre limité de structures récurrentes. Grâce à SpaCy, nous avons pu implémenter les règles permettant de reconnaître ces structures, et ce sous plusieurs formes. Il s'agit des patrons morphosyntaxiques, des expressions régulières, des règles décrivant les caractéristiques morphologiques de *token*, et finalement des combinaisons de *token* et de sous-*token*.

2.3.1.3 Règles à la base de l'analyse syntaxique

La plupart des expressions à repérer sont des syntagmes nominaux. Dans un arbre syntaxique, un tel syntagme se présente sous forme du sous arbre d'un mot clé. Nous élaborons donc une liste de mots clés à partir du corpus d'entraînement. L'analyse syntaxique se fait avec SpaCy et nous récupérons les sous arbres contenant ces mots clés. Ces sous arbres sont ensuite nettoyés afin d'éliminer les branches redondantes. Ce nettoyage est effectué à l'aide des filtres lexicaux et morphosyntaxiques. Finalement, ces arbres nettoyés constituent nos prédictions. Le plus grand avantage de ce type de règle est sa flexibilité. Ceci est surtout utile pour les catégories qui n'ont pas ou peu de contraintes morphologiques. Au lieu de récupérer un nombre défini de modifieurs d'un tel mot clé, nous pouvons

[3] https://spacy.io/
[4] http://agence-prd.ansm.sante.fr

prendre tous les modifieurs qui y sont attachés, puis enlever ceux qui ne sont pas pertinents. Ainsi nous évitons d'élaborer les règles trop précises seulement pour repérer quelques structures rares.

2.3.2 Résultats

	TP	FP	FN	Precision	Recall	F1
pathologie	61	386	105	0,1365	0,3675	0,1990
sosy	0	0	1279	0,0000	0,0000	0,0000
Overall	61	386	1384	0,1365	0,0422	0,0645

Table 3: résultats de la sous tâche 1

	TP	FP	FN	Precision	Recall	F1
anatomie	238	537	883	0,3071	0,2123	0,2511
dose	19	62	33	0,2346	0,3654	0,2857
examen	368	397	449	0,4810	0,4504	0,4652
mode	34	31	55	0,5231	0,3820	0,4416
moment	36	73	129	0,3303	0,2182	0,2628
substance	93	25	220	0,7881	0,2971	0,4316
traitement	88	247	216	0,2627	0,2895	0,2754
valeur	159	89	273	0,6411	0,3681	0,4676
Overall	1035	1461	2258	0,4147	0,3143	0,3576

Table 4 : résultats de la sous tâche 2

Les erreurs proviennent principalement des catégories « anatomie » et « examen », ce qui correspond à la contrainte de non chevauchement d'entités de SpaCy. Il est attendu que les termes anatomiques soient présents dans les syntagmes « examen » « traitement » et « pathologie », alors que ce genre d'annotations est difficile à gérer avec SpaCy.

Par ailleurs, un type d'erreurs fréquent est la frontière des syntagmes récupérés avec les règles syntaxiques. Elles permettent de trouver les syntagmes les plus longs, cependant ce n'est pas toujours ce qui est demandé. Par exemple, les adjectifs suivants un mot clé sont gardés, mais avec peu de contrôle du vocabulaire. Il est possible que les adjectifs « non médicaux » soient pris en compte. De plus, nous utilisons des filtres lexicaux et morphosyntaxiques pour nettoyer les sous arbres. Les erreurs de parsing morphosyntaxique peuvent donc également conduire à un mauvais découpage de syntagme.

3 Conclusion

Participer à la campagne DEFT 2020, nous a permis de tester plusieurs méthodes de calcul de similarité dont les résultats prometteurs pourront être utilisés directement à EDF Commerce et à d'autres entités du groupe EDF (tâches 1 et 2). Nous travaillons également régulièrement sur l'évaluation des outils d'extraction d'entités nommées, comme SpaCy. Les données médicales du défi, bien que différentes de nos données métiers, sont pour nous l'occasion d'évaluer ces outils, de connaître leurs avantages mais également leurs limites.

Références

CARDON R, GRABAR N. Identification of parallel sentences in comparable monolingual corpora from different registers. In: *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*. 2018. p. 83-93

CARDON R, GRABAR N, GROUIN C, HAMON T (2020). Présentation de la campagne d'évaluation DEFT 2020 : similarité textuelle en domaine ouvert et extraction d'information précise dans des cas cliniques. In: Actes de DEFT.

CER, D, YANG, YINFEI, KONG, SHENG-YI, ET AL. Universal sentence encoder. arXiv preprint arXiv:1803.11175, 2018.

DICE, L. R. (1945). Measures of the amount of ecologic association between species. Ecology, 26(3), 297-302.

HALL, M., FRANK, E., HOLMES, G., PFAHRINGER, B., REUTEMANN, P., & WITTEN, I. H. (2009). The WEKA data mining software: an update. ACM SIGKDD explorations newsletter, 11(1), 10-18.

NIVRE, J., DE MARNEFFE, M. C., GINTER, F., GOLDBERG, Y., HAJIC, J., MANNING, C. D., ... & TSARFATY, R. (2016, May). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)* (pp. 1659-1666).

URIELI, A. (2013). *Robust French syntax analysis: reconciling statistical methods and linguistic knowledge in the Talismane toolkit* (Doctoral dissertation).