Participation d'EDF R&D à DEFT 2018

Philippe Suignard, Lou Charaudeau, Manel Boumghar, Meryl Bothua, Delphine Lagarde
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 2018. Notre équipe a participé aux deux premières tâches : classification des tweets en transport/non-transport (Tâche T1) et détection de la polarité globale des tweets (Tâche T2). Nous avons utilisé 3 méthodes différentes s'appuyant sur Word2Vec, CNN et LSTM. Aucune donnée supplémentaire, autre que les données d'apprentissage, n'a été utilisée. Notre équipe obtient des résultats très corrects et se classe 1ère équipe non académique. Les méthodes proposées sont facilement transposables à d'autres tâches de classification de textes courts et peuvent intéresser plusieurs entités du groupe EDF.

ABSTRACT

Here the title in English.
This paper describes the participation of EDF R&D at DEFT 2018 evaluation campaign. Our team participated in the first two tasks: classification of tweets in transport / non-transport (Task T1) and detection of the overall polarity of tweets (Task T2). We used 3 different methods based on Word2Vec, CNN and LSTM. No additional data other than the training data was used. Our team gets very correct results and ranks 1st non-academic team. The proposed methods are easily transferable to other short text classification tasks and may interest several entities of the EDF group.

MOTS-CLÉS : Tweet, Polarité, Word2Vec, LSTM, CNN.
KEYWORDS: Tweet, Polarity, Word2Vec, LSTM, CNN.

1 Introduction

Plusieurs éléments nous ont motivés à participer à l'édition 2018 du défi DEFT (Paroubek, 2018) :

- EDF R&D travaille en appui de la Direction EDF Commerce à la mise en œuvre d'une chaine de récupération des tweets, de classification et d'attribution d'une polarité (travail en cours qui s'apparente à la tâche 2).
- Dans la phase de récupération des tweets, nous avons une problématique similaire à la tâche 1, non pas sur la distinction des tweets « transport » / « inconnu », mais EDF « Electricité de France » / EDF « Equipe de France ».
- La volumétrie importante des tweets mis à disposition pour ce concours nous permettait d'envisager des méthodes de type Machine Learning.

Participer à DEFT était l'occasion de travailler sur plusieurs méthodes de classification dont les résultats contribueront directement à EDF Commerce et à d'autres entités du groupe EDF.

2 Description des méthodes utilisées

Les trois méthodes que nous avons proposées partagent la même approche à savoir d'entrainer deux modèles chacune (un pour la tâche 1 et un autre pour la tâche 2). Le premier modèle permet de discriminer les tweets entre les catégories « transport » et « inconnu ». Le second modèle discrimine les tweets selon leur polarité.

Deux corpus étaient fournis pour l'apprentissage « batch_b », composé d'environ 14 000 tweets, et « simple », composé d'environ 54 000 tweets. Comme tous les tweets de « batch_b » étaient étiquetés « inconnu », nous avons préféré ne pas les utiliser.

Aucune donnée externe supplémentaire n'a été utilisée. Pour entrainer nos méthodes, nous avons découpé le corpus « simple » en deux parties : 80% étant réservé pour l'apprentissage et 20% ont servi pour les tests. Quelques tweets vides ou erronés ont été éliminés.

2.1 Méthode 1 basée sur Word2Vec

La méthode 1 est inspirée de (Xing, 2017), elle consiste à entrainer un modèle Word2Vec à la fois sur les mots des tweets et sur leur catégorie. A l'aide du modèle Word2Vec ainsi entrainé, une série de descripteurs sont calculés pour chaque tweet pour ensuite entrainer un classifieur. La suite de ce paragraphe présente les différentes étapes suivies.

2.1.1 Nettoyage des tweets

Les tweets sont pré-traités de la manière suivante :

- Passage des mots en minuscule ;
- Suppression des lettres redoublées, ce qui transforme « cooooooool » en « col », « arrêt » en « arêt », « mdrrrr » et « mddddrrr » en « mdr ». Cela permet de réduire le vocabulaire et de s'affranchir d'une certaine partie des fautes d'orthographe ;
- Transformation des URL par « HTTP » ;
- Changement des dates du type 01/02/2016 par « DATE » ;
- Changement des heures du type 12h30 ou 12:20 par « HEURES » ;
- Changement des durées du type 12 min, 12mn, 12 minutes par « DUREE » ;
- Transformation des smiley par « SMILEYHAPPY », « SMILEYSAD », « SMILEYWINK », « SMILEYCRYING », etc. sans toutefois être exhaustif ;
- Suppression des caractères de mention et de hashtag : « @ratp » devient « ratp » et « #sncf » devient « sncf ».

2.1.2 Word2Vec

On ne présente plus la méthode Word2Vec (Mikolov, 2013) qui consiste à transformer des mots en vecteurs.

La 1ère méthode pour classifier les tweets va s'appuyer sur Word2Vec, avec comme idée centrale, le fait d'entrainer un modèle en mélangeant les mots d'un tweet avec la catégorie de ce tweet. Pour entrainer le modèle pour la tâche 1, le tweet « Les gars qui puent des aisselles dans le bus c'est vous » ayant pour catégorie « TRANSPORT », devient :

- <transport> Les gars qui puent des aisselles dans le bus c'est vous <transport>

Pour entrainer le modèle pour la tâche 2, il devient :

- <negatif> Les gars qui puent des aisselles dans le bus c'est vous <negatif>

Ainsi, un mot fréquemment utilisé dans la catégorie « transport » et très connoté « transport » aura tendance à être d'avantage similaire (au sens de Word2Vec) du mot « transport ». Il en va de même pour les catégories « inconnue », « positif », « négatif », « neutre » et « mixte ».

Les paramètres retenus pour entrainer les modèles Word2Vec sont : Skip-Gram, un voisinage de 5 mots à droite et à gauche, une couche cachée de taille 200, un softmax hiérarchique, une fréquence minimale des mots de 30, les mots de un ou deux caractères sont éliminés et 1000 itérations pour entrainer le modèle.

Une stop-liste a été utilisée pour éliminer les mots ayant peu de valeur ajoutée pour la problématique concernée.

2.1.3 Calcul des descripteurs

Pour chaque tweet du corpus d'entrainement, sont calculés les descripteurs suivants :

- Similarité **vectorielle moyenne** : calcul de la moyenne des vecteurs mots du tweet et calcul de la similarité entre ce vecteur mot et les vecteurs mots des catégories « transport » et « inconnu » pour la tâche 1 (soit 2 descripteurs pour T1 et 4 pour T2).
- Similarité **distributionnelle** : calcul de la distribution des N plus proches voisins des mots du tweet (au sens de Word2Vec), calcul de la distribution des N plus proches voisins des catégories « transport » et « inconnu » pour la tâche 1, puis calcul de la similarité entre ces deux distributions. Ce qui fait 2 descripteurs pour T1 et 4 pour T2.
- Similarité **centrale** : calcul préalable des vecteurs moyens pour chaque catégorie du corpus d'apprentissage (un vecteur pour la catégorie « transport » et un autre pour « inconnu », par exemple), puis calcul de la similarité entre chaque tweet et ces tweets moyens, soit 2 descripteurs pour T1 et 4 pour T2.
- **Moyenne des similarités des mots** : calcul de la similarité entre chaque mot du tweet et les différentes catégories (« transport » et « inconnu » pour la tâche 1), puis calcul de la moyenne de ces similarités, soit 2 descripteurs pour T1 et 4 pour T2.
- **Maximum des similarités des mots** : même chose que précédemment, mais en retenant uniquement le maximum des similarités, soit 2 descripteurs pour T1 et 4 pour T2.
- **Minimum des similarités des mots** : même chose que précédemment, mais en retenant uniquement le minimum des similarités, soit 2 descripteurs pour T1 et 4 pour T2.
- **Ecart types des similarités des mots** : même chose que précédemment, mais en retenant uniquement les écarts types des similarités, soit 2 descripteurs pour T1 et 4 pour T2.

Pour la tâche 2, comme on cherche à détecter la présence de mots positifs et de mots négatifs au sein d'un même tweets, notamment pour détecter les tweets de la catégorie « mixte », on ajoute les descripteurs suivants :

- **Maximum des ruptures de similarité entre deux mots successifs** ;
- **Minimum des ruptures de similarité entre deux mots successifs** ;
- **Maximum des ruptures de similarité entre deux mots du tweet** ;
- **Minimum des ruptures de similarité entre deux mots du tweet**, soit 4x4 = 16 descripteurs supplémentaires pour T2;

Ce qui fait au total 14 descripteurs par tweet pour la tâche T1 et 44 pour la tâche 2.

2.1.4 Entrainement d'un classifieur

Les descripteurs ont été calculés pour chaque tweet du corpus d'apprentissage puis convertis au format ARFF pour être utilisées au sein du logiciel WEKA (Hall, 2009). Plusieurs classifieurs ont été testés : « Régression Logistique » et « Random Forest ».

Les classifieurs ainsi entrainés ont été appliqués sur les 7816 tweets du corpus d'évaluation. Un seul « run » a été proposé pour la tâche 1 (avec Random Forest, l'écart entre les deux méthodes n'étant pas significatif) et deux « run » ont été proposés pour la tâche 2, respectivement avec les méthodes « Régression Logistique » et « Random Forest ».

2.2 Méthode 2 : LSTM + prétraitement

2.2.1 Annotation des tweets pour la détection de polarité

Nous avons ajouté automatiquement des étiquettes de polarité pour aider l'apprentissage des modèles LSTM et CNN lors de la tâche 2. Nous avons ainsi constitué manuellement deux lexiques, l'un contenant des mots de polarité positive, l'autre contenant des mots de polarité négative. Ces mots sont issus du corpus de tweets. Notre lexique de polarité positive comprend 121 entrées et celui de polarité négative 291. Ils intègrent tous deux des émoticônes. Ils ne prétendent pas à l'exhaustivité mais les annotations qui en découlent ajoutent des métadonnées pour les modèles LSTM et CNN. Ces annotations sous la forme d'étiquettes <POSITIF> ou <NEGATIF> sont ajoutées automatiquement dans les tweets qui comprennent un mot appartenant à l'un ou l'autre des lexiques. Ces prétraitements ont été appliqués sur le corpus d'entraînement et sur le corpus de test. En voici deux exemples :

> 699945537311793152 préavis de grève <**NEGATIF**> à la @sncf pour ce jeudi en rhônealpes et jusqu au 23 février. risques de perturbations <**NEGATIF**> sur le trafic <**NEGATIF**> ter

> 715780410165301248 j suis trop bien <**POSITIF**> dans le bus aek ma musique et le chaufage à côté ☺ **<POSITIF>**

Cet ajout de polarité nous a fait gagner environ 5% en « accuracy » sur le corpus d'apprentissage.

2.2.2 Entrainement de 2 LSTM

Pour cette deuxième méthode, nous avons choisi d'utiliser des réseaux de neurones, basés sur des LSTM (Hochreiter, 1997), qui obtiennent de bonnes performances sur des tâches de classification textuelle (Wenpeng, 2017). Les LSTM étant plus performants lorsque la classification à réaliser est concentrée sur une seule thématique, nous avons choisi d'entrainer un réseau par tâche plutôt qu'un seul global. Nous avons ainsi entrainé 2 réseaux de neurones séparés pour chaque tâche de classification : le premier avait pour objectif de séparer les tweets entre « INCONNU » et « TRANSPORT » et le second d'attribuer une polarité aux tweets identifiés dans la catégorie « TRANSPORT »

Chaque réseau a été construit en associant une première couche d'embedding, un LSTM puis une couche dense en sortie de manière à combiner les sorties du LSTM. Pour chaque tâche, nous avons réalisé une exploration des paramètres suivants :

- Taille du padding des phrases d'entrée ;
- Taille d'embedding ;
- Taille de la couche cachée du LSTM.

Pour chaque paramètre, nous avons sélectionné une plage de variation, puis entrainé 10 modèles sur chaque jeu de paramètres différents.

Les performances moyennes des modèles ont été estimées pour chaque jeu de paramètres selon 2 critères :

- **L'accuracy du modèle sur le jeu de validation** : pourcentage de bonnes classifications réalisées par le modèle après entrainement
- **La « loss » sur le jeu de validation** : somme des erreurs réalisées par le modèle sur le jeu de validation

Pour la première tâche, chaque modèle a été entrainé sur une sélection aléatoire de 80% des tweets du jeu de tweets d'entrainement puis validé sur les 20% restants. Pour la seconde tâche, les modèles ont été entrainés sur 80% des tweets étiquetés « TRANSPORT » puis validés sur les 20% restants. L'apprentissage a été réalisé sur des mini-batch de 32 échantillons.

Les meilleures performances ont été obtenues pour les paramètres suivants :

- Première tâche : embedding de taille 150, padding de taille 40 et couche cachée du LSTM de 60 neurones
- Deuxième tâche : embedding de taille 300, padding de taille 40 et couche cachée du LSTM de taille 100

Si le choix de paramètres pour la première tâche est relativement classique, la tâche de classification étant dans un cadre simple, il est plus complexe pour la deuxième tâche : la catégorisation plus complexe pousse à augmenter le nombre de neurones du réseau, alors que le nombre de données d'apprentissage est plus réduit, ce qui augmente le risque de surapprentissage. Nous avons ainsi choisi d'orienter les modèles vers des réseaux plus grands, tout en ajoutant un dropout de 20% pour contrer le surapprentissage.

2.3 Méthode 3 : CNN + prétraitement

2.3.1 Annotation des tweets pour la détection de polarité

Nous avons appliqué les mêmes prétraitements que ceux décrits en 2.2.1.

2.3.2 Entrainement du CNN

Enfin, pour la troisième méthode, nous avons choisi d'utiliser des réseaux de neurones, basés sur des CNN (LeCun, 1998). Le caractère hiérarchique des CNN en font de bons candidats pour traiter des tâches de classification de textes. En effet, ces structures ont été fréquemment utilisées dans la littérature pour des tâches de classification notamment d'analyse de sentiment (Dauphin, 2016).

Les CNN ont été entraînés uniquement pour la deuxième tâche, c'est-à-dire l'attribution d'une polarité aux tweets identifiés dans la catégorie « TRANSPORT ».

Pour cette tâche nous avons choisi d'explorer les paramètres ci-dessous :

- Taille du padding des phrases d'entrée
- Taille d'embedding
- Taille de la fenêtre de filtre
- Taille de la fenêtre de pooling

De la même manière que pour les LSTM, les performances moyennes des modèles ont été estimées pour chaque jeu de paramètres selon 2 critères :

- L'accuracy du modèle sur le jeu de validation
- La loss sur le jeu de validation

Aussi, pour l'entrainement des modèles, ces derniers ont été entrainés sur 80% des tweets étiquetés « TRANSPORT » puis validés sur les 20% restants. L'apprentissage a été réalisé sur des mini-batch de 32 échantillons.

Les meilleures performances ont été obtenues pour les paramètres suivants :

- Paramètres retenus pour la deuxième tâche : embedding de taille 300, padding de taille 40 et une taille de filtre et de pooling de 5 mots

3 Résultats obtenus

Les tables 1 et 2 récapitulent les résultats obtenus par nos méthodes sur les tâches 1 et 2. F-Mesure est la F-Mesure obtenu par nos méthodes, F-Mesure moyenne, est la moyenne des F-Mesure de tous les run de tous les participants et F-Mesure Max est le maximum des F-Mesures des participants.

Tâche 1, nom de la méthode :	F-Mesure	F-Mesure moyenne	F-Mesure Max
Word2Vec + Classifieur Random Forest	**0,90286**	0,88813	0,90785
LSTM + Prétraitements	0,90124	0,88813	0,90785

TABLE 1 – Résultats de notre participation à la campagne Deft 2018 pour la tâche 1.

Tâche 2, nom de la méthode :	F-Mesure	F-Mesure moyenne	F-Mesure Max
Word2Vec + Classifieur Regression Logistique	0,7435	0,7293	0,82288
Word2Vec + Classifieur Random Forest	0,73969	0,7293	0,82288
LSTM + Prétraitements	**0,80249**	0,7293	0,82288
LSTM2(*) + Prétraitements	0,79957	0,7293	0,82288
CNN + Prétraitements	0,80067	0,7293	0,82288

TABLE 2 – Résultats de notre participation à la campagne Deft 2018 pour la tâche 2.

(*) : LSTM2 est la même méthode que LSTM mais en restreignant la taille du vocabulaire.

De ces résultats, nous tirons les enseignements suivants :

- EDF R&D se place 5ème sur la tâche 1 (sur 11 équipe) et 4ème sur la tâche 2 (sur 12 équipes).
- Tous nos « run » ont une « F-mesure » au-dessus de la moyenne des F mesures.
- Nous sommes classés 1er acteur non académique.
- Sur la tâche 1, c'est la méthode basée sur Word2Vec qui obtient, de très peu, le meilleur résultat.
- Sur la tâche 2, c'est la méthode basée sur les LSTM qui obtient les meilleurs résultats.

Selon notre point de vue, notre participation à ce concours est positive, car elle nous a permis de tester plusieurs méthodes de classification de textes courts.

4 Conclusion

Participer à la campagne DEFT 2018, nous a permis de tester 3 méthodes de classifications de textes courts basées sur Word2Vec, LSTM et CNN. Aucune donnée supplémentaire, autre que les données d'apprentissage, n'a été utilisée. Les résultats obtenus sont satisfaisants. Les méthodes que nous avons mises en œuvre sont facilement transposables à d'autres tâches de classification de textes courts et peuvent intéresser plusieurs entités du groupe EDF.

Références

DAUPHIN Y. N., FAN, A., AULI M., & GRANGIER D. (2016). Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083*.

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.

HOCHREITER .S, AND SCHMIDHUBER J. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

LECUN Y., BOTTOU L., BENGIO Y., & HAFFNER P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

MIKOLOV T., SUTSKEVER I., CHEN K., CORRADO G. S., & DEAN J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).

MIKOLOV T., CHEN K., CORRADO G., & DEAN J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

PAROUBEK P., GROUIN C., BELLOT P., CLAVEAU V., ESHKOL-TARAVELLA I., FRAISSE A., JACKIEWICZ A., KAROUI J., MONCEAUX L., TORRES-MORENO J.M. DEFT2018 : recherche d'information et analyse de sentiments dans des tweets concernant les transports en Île de France. In: Actes de DEFT. Rennes, France.

WENPENG Y., KATHARINA K., MO Y., HINRICH S. (2017) Comparative study of CNN and RNN for language processing. ARXIV PREPRINT ARXIV :1702.01923.

XING L., & PAUL M. J. (2017). Incorporating Metadata into Content-Based User Embeddings. In Proceedings of the 3rd Workshop on Noisy User-generated Text (pp. 45-49).