Détection des mots non-standards dans les tweets avec des réseaux de neurones

Tian Tian[1, 2] Marco Dinarelli[1] Pedro Cardoso[2] Isabelle Tellier[1]
(1) Lattice (UMR 8094), CNRS, ENS Paris, Université Sorbonne Nouvelle,
PSL Research University, USPC, 1 rue Maurice Arnoux, 92120 Montrouge, France
(2) Synthesio, 8-10 rue Villedo, 75001 Paris
ttian@synthesio.com, marco.dinarelli@ens.fr, pedro@synthesio.com,
isabelle.tellier@univ-paris3.fr

RÉSUMÉ

Dans cet article, nous proposons un modèle pour détecter dans les textes générés par des utilisateurs (en particulier les tweets), les mots non-standards à corriger. Nous utilisons pour cela des réseaux de neurones convolutifs au niveau des caractères, associés à des "plongements" (embeddings) des mots présents dans le contexte du mot courant. Nous avons utilisé pour l'évaluation trois corpus de référence. Nous avons testé différents modèles qui varient suivant leurs plongements pré-entrainés, leurs configurations et leurs optimisations. Nous avons finalement obtenu une F1-mesure de 0.972 en validation croisée pour la classe des mots non-standards. Cette détection des mots à corriger est l'étape préliminaire pour la normalisation des textes non standards comme les tweets.

ABSTRACT

Detecting non-standard words in tweets with neural networks

In this paper, we propose a model for detecting non-standard words to be corrected in user-generated texts (such as tweet). We use character-level convolutional neural networks, associated with word embeddings for the current word's context. We conducted our evaluations with three reference corpora. We tested different pretrained word embedding models, different configurations and different optimizations for our various models. We finally obtained a F1-measure of 0.972 in cross validation for the non-standard words class. This non-standard word detection is a first step towards the normalisation of tweets.

MOTS-CLÉS : mots non-standards, réseaux de neurones, modèle convolutionnel, plongements.

KEYWORDS: non-standard words, neural networks, convolutional model, embeddings.

1 Introduction

Les contenus générés par des utilisateurs sur le web, par exemple via les tweets, sont devenus une source de données importante. Ces textes, sous la forme de petits messages d'au maximum 140 caractères, véhiculent souvent des opinions sur l'actualité ou sur des produits de consommation. Ils sont donc particulièrement intéressants à analyser automatiquement dans le cadre de la fouille d'opinion. Cependant, les outils traditionnels de Traitement Automatique des Langues (TAL) se comportent

rarement bien face à ces textes, courts et souvent écrits dans une langue non standard. Par exemple, le Stanford POS Tagger, étiqueteur morpho-syntaxique appris avec le Penn TreeBank [1] (Toutanova & Manning, 2000) atteint 96.86% d'exactitude sur les textes journalistiques, mais seulement 81.3% sur les tweets (Ritter *et al.*, 2011). Ce phénomène se retrouve aussi pour d'autres tâches de TAL, comme l'analyse syntaxique (Plank *et al.*, 2014; Kong *et al.*, 2014). La *normalisation de texte* est la tâche qui consiste à transformer *a minima* un texte "non standard" pour le faire se rapprocher le plus possible d'un texte "standard" sans modifier son sens. C'est devenu une étape de prétraitement nécessaire pour certains textes, avant d'essayer de leur appliquer d'autres étapes de TAL (Han *et al.*, 2013). Elle opère pour l'instant essentiellement au niveau lexical, par l'identifiaction et le remplacement des mots (ou "tokens") non standards. Beaucoup de travaux montrent qu'une telle normalisation appliquée sur des tweets peut améliorer le résultat de l'extraction des entités nommées qu'ils contiennent (Nguyen *et al.*, 2016; Liu *et al.*, 2012). Dans certains cas, la détection des unités non-standards peut à elle seule améliorer ce résultat (Li & Liu, 2015).

Notre travail dans cet article s'inspire de certaines expériences de (Li & Liu, 2015). Il vise au final à améliorer l'extraction des entités nommées dans les tweets après une phase de normalisation. Nous nous focalisons en particulier sur le pré-traitement lexical permettant de rendre les tweets un peu mieux formés, en remplaçant les mots non-standards qui y figurent par une forme standard. Nous essayons ici, dans un premier temps, de détecter dans un tweet ces mots non-standards avant de proposer une liste de candidats pour les corriger.

Nous proposons pour cela un modèle de réseau de neurones convolutifs au niveau des caractères, associé à des "plongements" de mots, pour déterminer si un mot est non-standard (à corriger). Nous avons utilisé pour nous évaluer des corpus de tweets en anglais dans lesquels les mots non standards ont été manuellement corrigés et avons mené diverses expériences avec divers protocoles (en validation croisée, ou avec un autre corpus d'évaluation).

Cet article est organisé de la manière suivante :

— la section 2 détaille la tâche de normalisation des tweets ainsi que l'état de l'art ;
— la section 3 présente les corpus annotés utilisés dans cet article ;
— la section 4 explique en détail le modèle de réseau de neurones convolutifs utilisé ;
— la section 5 contient les résultats obtenus avec les différentes configurations ;
— la section 6 conclut sur ces expériences et évoque quelques perspectives pour l'avenir.

2 Etat de l'art

2.1 Notion de mots non-standard

Les mots non-standards (non-standard words, NSW) sont des mots à corriger dans le contexte où ils apparaissent par un mot présent dans un vocabulaire. Ils peuvent contenir des fautes d'orthographe (comme "dis" pour "this" en anglais), des acronymes (comme "lol" pour "laughing out loud"), des abréviations (comme "u" pour "you"), etc. Cette notion de NSW se distingue de celle de mots hors vocabulaire (out-of-vocabulary, OOV). En effet, les OOV sont des mots absents d'un dictionnaire de référence (par exemple celui de GNU Aspell pour l'anglais [2]), mais ne sont pas nécessairement pour autant mal écrits et à corriger : les entités nommées ou noms de produits sont souvent OOV.

1. Stanford Pos Tagger : http ://nlp.stanford.edu/software/tagger.shtml
2. GNU Aspell : http ://aspell.net/

Ainsi, un mot comme "xanax" (nom d'un médicament contre l'anxiété), ne se trouve pas dans les dictionnaires. Ce n'est cependant pas un NSW parce qu'il n'y a pas moyen de le corriger, il est déjà sous sa forme correcte. Les NSW des exemples précédents sont des OOV, mais il y a aussi des NSW qui figurent dans un vocabulaire, mais ne conviennent pas dans leur contexte, par exemple "wit" dans la phrase "I'll go wit you" (la forme correcte est "with") (Han, 2014). Le contexte joue donc un rôle très important pour ranger les mots dans ces différentes classes. Enfin, du bruit peut aussi exister dans les tweets, comme "bahahahaha" à la fin d'un message, pour exprimer un sentiment. Ce token est OOV parce qu'il n'est pas présent dans les dictionnaires usuels, mais il n'existe pas non plus de forme correcte correspondante. Il ne s'agit donc pas d'un NSW. Certaines fois, il est difficile de juger si un token est NSW ou SW comme "footie", "y'all" et "yous" (Baldwin *et al.*, 2015).

2.2 Normalisation des tweets

La normalisation lexicale de texte peut se décomposer en trois sous-tâches :

1. Trouver les NSW de ce texte. Dans un corpus de tweets, le taux de NSW est d'environ 10%. Cette sous-tâche se traduit souvent par une classification binaire (même si, comme nous venons de le voir, la situation est souvent plus compliquée à cause du bruit). Il s'agit donc à cette étape de décider, pour chaque token d'un texte, s'il est à corriger ou pas.
2. Pour un NSW dans un texte, proposer une liste de candidats pour le corriger. Si un token peut être considéré comme candidat à sa propre correction, alors cette sous-tâche peut être exécutée pour tous les tokens d'un texte, sans passer par la première sous-tâche comme dans (Jin, 2015) et (Mausam *et al.*, 2012). (Han & Baldwin, 2011) a proposé de générer les corrections des NSW en se basant sur les similarités morpho-phonétiques. (Supranovich & Patsepnia, 2015) a, lui, fait appel à la distance d'édition (ou distance de Levenshtein) (Miller *et al.*, 2009).
3. Pour un NSW et une liste de candidats de correction proposée, trouver le candidat qui a la plus forte probabilité d'être la forme correcte du NSW. Cela peut être réalisé en calculant des mesures de similarité entre le NSW et chacun de ses candidats comme dans (Jin, 2015), ou par la distance d'édition (Miller *et al.*, 2009) comme (Supranovich & Patsepnia, 2015).

Pour la première sous-tâche, (Han & Baldwin, 2011; Supranovich & Patsepnia, 2015; Leeman-Munk *et al.*, 2015) ont utilisé un classifieur pour détecter les NSW. (Supranovich & Patsepnia, 2015) a utilisé les champs conditionnels aléatoires (Conditional Random Fields, CRF) tandis que (Leeman-Munk *et al.*, 2015) ont utilisé un réseau de neurones pour le faire. (Berend & Tasnádi, 2015) a quant à lui exploité les CRF pour la classification et la correction des NSW.

La correction de tweets a fait l'objet d'un challenge dans le workshop WNUT (Workshop on Noisy User-generated Text (Baldwin *et al.*, 2015)). Les meilleurs résultats ont été obtenus par (Leeman-Munk *et al.*, 2015), qui ont atteint un résultat de $0,98$ de F1-mesure pour la classe NSW sur le corpus de développement du workshop, et le même résultat sur le corpus d'évaluation. Dans cet article, nous essayons de reprendre la méthode qu'ils ont utilisée pour détecter dans des tweets les mots non-standards (NSW), considéré comme une tâche de classification binaire.

3 Les corpus annotés

Nous avons utilisé trois corpus annotés : 2577 tweets provenant de (Li & Liu, 2014) d'une part, les corpus de développement et d'évaluation du workshop WNUT d'autre part (Han & Baldwin, 2011;

Baldwin *et al.*, 2015). Le tableau 1 fournit les statistiques de base de ces trois corpus.

nom de corpus	nombre de tweets	# tokens	# NSW	# SW	taille de vocabulaire
2577 tweets	2577	44385	3942	40443	8270
WNUT dev	2950	40560	4274	36286	15108
WNUT eval	1698	29421	2776	26645	10765

TABLE 1 – Corpus statistiques

Ces trois corpus sont tokenisés puis annotés avec la correction manuelle de NSW. Pour la convenance des annotateurs, les annotations (les corrections de tokens) sont en minuscules. Nous avons donc utilisé uniquement l'annotation pour distinguer les NSW des SW pour notre tâche de détection des NSW : si la forme de surface d'un token dans le corpus est différente de sa correction (la différence de majuscules et de minuscules sont ignorées), ce token est considéré comme un NSW.

D'après les statistiques du tableau 1, les mots non-standards sont minoritaires, ils constituent respectivement 8.9%, 10.5% et 9.4% du nombre total des tokens de ces trois corpus.

4 Les modèles de réseaux de neurones convolutifs

Les modèles de réseaux de neurones convolutifs (CNN) introduits dans (Collobert *et al.*, 2011) sont utilisés pour extraire les propriétés de haut niveaux des vecteurs de mots. Dans (Ma & Hovy, 2016), un modèle convolutif similaire au niveau des caractères est aussi utilisé pour représenter des propriétés plus fines. Dans cet article, nous proposons un modèle de réseau de neurones convolutifs au niveau des caractères, associé à des plongements. La figure 1 montre le détail de la représentation du mot "Mercure" par la convolution des caractères comme dans (Ma & Hovy, 2016).

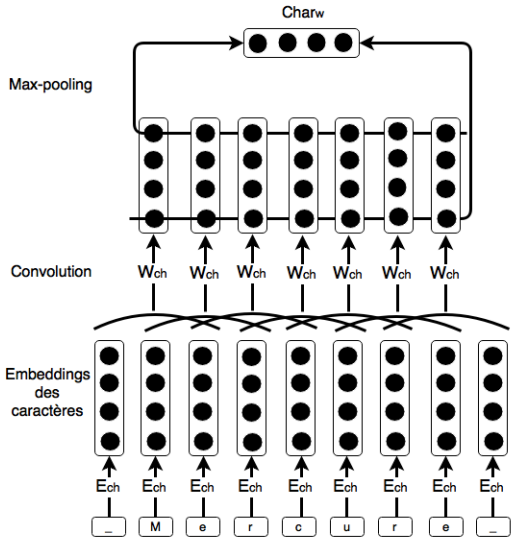

FIGURE 1 – structure du CNN sur les caractères utilisé

Dans cette figure, les E_{ch} désignent les plongements (embeddings) des caractères (un vecteur pour

représenter un caractère, de taille quatre sur le dessin). Les symboles "_" avant le "M" et après le "e" du mot servent à le compléter pour la fenêtre de taille trois autour d'un caractère. Les W_{ch} sont les poids de la matrice correspondant à un triplet de caractères. Enfin le max-pooling prend uniquement la valeur la plus élevée de chaque composante du vecteur pour arriver à une représentation du mot "Mercure" ($Char_w$). Dans notre modèle, la taille de ce vecteur est en réalité 300 (comme la configuration de modèle word2vec de Google).

Nous avons ensuite ajouté à la représentation du mot cible ainsi obtenue son contexte dans une fenêtre couvrant deux mots avant et deux mots après lui, représentés avec des plongements de mots classiques ($Emb(W_i)$, décrits plus loin). Les débuts et la fin de tweets sont marqués par <s> <s> et </s> </s> pour constituer une suite de mots de taille constante (5-grammes). Ainsi, un tweet de N mots sera présenté par N 5-grammes. Par exemple, pour un tweet avec six tokens : "I txtd you last night." il y a six 5-grammes qui représentent cette séquence de texte : "<s> <s> I txtd you", "<s> I txtd you last", "I txtd you last night", "txtd you last night .", "you last night . </s>" et "last night . </s> </s>". La structure complète du modèle est illustrée dans le schéma 2.

Emb(W_{-2})	Emb(W_{-1})	Emb(W_0)	Emb(W_1)	Emb(W_2)	représentation du mot
Dropout(0.2)					
Couche cachée (dimension = 128, activation = ReLU)					
Dropout(0.2)					
Couche de sortie (dimension = 2, activation = softmax), loss = categorical_crossentropy					

TABLE 2 – Structure du réseau

5 Experiences et résultats

Notre baseline est une méthode simple par dictionaire induit sur les données d'entraînement : pour chaque token des données d'apprentissage, s'il est au moins une fois marqué comme NSW, il est stocké dans un ensemble (la casse est toujours ignorée comme convenu). Dans la phase de test, si le token est dans cet ensemble, il sera taggé comme NSW, et comme SW sinon. Cette baseline reconnaît donc uniquement les NSW déjà présents dans les données d'apprentissage ; les autres tokens des données de test seront taggés comme SW.

Nous avons procédé à diverses expériences avec notre réseau de neurones convolutif. Le tableau 3 montre les résultats de la validation croisée en 5 blocs avec le corpus WNUT dev et avec le corpus WNUT dev plus le corpus de 2577 tweets. Le tableau 4 montre les résultats d'évaluation obtenus sur WNUT test par un modèle appris sur WNUT dev et sur WNUT dev plus 2577 tweets. Ces derniers résultats sont donc comparables avec les travaux de (Leeman-Munk *et al.*, 2015).

Les conditions d'expérimentation sont les suivantes :

— Logiciel d'implémentation : Keras [3], Theano [4]

— Initialisation des paramètres : Xavier initialisation (Glorot & Bengio, 2010)

3. Keras : https ://keras.io/

4. Theano : http ://deeplearning.net/software/theano/

— Plongements pré-entrainés des mots ($Emb(W_i)$) : Google word2vec (Mikolov *et al.*, 2013) versus *interne*.

Le modèle *interne* est appris avec word2vec sur des textes plus longs et mieux formés que les tweets (news, forums, blogs, etc.) mais portant sur des domaines similaires (produits de luxe, automobiles et musique). La taille du vocabulaire de ce modèle est d'environ deux millions de tokens différents, tandis qu'il y a trois millions de tokens différents dans le modèle word2vec de Google. Certaines entités nommées comme les dates, heures, unités de longueur, de volumes, etc. sont remplacées par un token artificiel (par exemple "__DATE__" pour les dates). Les tokens qui ne varient que suivant la casse sont aussi normalisés par leur forme la plus courante. Par exemple, le token "nissan" apparaît sous trois formes : tout en minuscule, tout en majuscule et "Nissan". La forme "Nissan" a le plus grand nombre d'occurrences dans le corpus *interne*, les formes "NISSAN" et "nissan" sont donc remplacées par "Nissan". Le modèle de word2vec *interne* est ainsi appris sur un vocabulaire restreint. En revanche, le modèle word2vec de Google (Mikolov *et al.*, 2013) a gardé toutes les formes pour tous les tokens. Donc les tokens "NISSAN", "Nissan" et "nissan" y ont des représentations vectorielles différentes (parce que leurs contextes sont aussi différents).

Nous avons testé les configurations suivantes :

— Corpus d'apprentissage : WNUT dev versus WNUT dev + 2577 tweets

— Optimisers : RMS prop (Tieleman & Hinton, 2012), adadelta (Zeiler, 2012), adagrad (Duchi *et al.*, 2011), sgd (Bergstra & Bengio, 2012)

Dans le tableau 3, les expériences en validation croisée avec 5 plis sont effectuées d'abord avec seulement le corpus de WNUT dev en premières lignes, puis sur les corpus 2577 tweets et WNUT dev mélangés (les dernières lignes). Pour les expériences sur le corpus WNUT eval, le modèle est appris avec seulement le corpus de WNUT dev pour les premières lignes, puis avec les corpus 2577 tweets et WNUT dev ensemble.

Training data	Word2vec	Optimizer name	précision NSW	rappel NSW	f1 measure NSW
WNUT dev	baseline		0.6382	0.5733	0.6092
	google	rmsprop	0.8859	0.7721	0.8242
		adadelta	0.8813	0.7976	0.8372
		adagrad	0.8941	0.7819	0.8341
		sgd	0.8823	0.8127	0.8459
	interne	rmsprop	0.9047	0.8013	0.8496
		adadelta	0.911	0.8022	0.853
		adagrad	0.9157	0.8062	0.8574
		sgd	0.9022	0.824	0.8612
WNUT dev+2577	baseline		0.3012	0.5388	0.3864
	google	rmsprop	0.9727	0.9501	0.9613
		adadelta	0.9813	0.9576	0.9693
		adagrad	0.9791	0.9566	0.9677
		sgd	0.9812	0.9597	0.9703
	interne	rmsprop	0.9808	0.9545	0.9674
		adadelta	0.9841	0.9620	**0.9729**
		adagrad	0.9876	0.9583	0.9727
		sgd	0.9853	0.9602	0.9725

TABLE 3 – Résultat en validation croisée

Nous pouvons observer dans ces résultats :

1. sans surprise, plus le corpus d'entraînement est grand, meilleurs sont les résultats : WNUT

Training data	Word2vec	Optimizer name	précision NSW	rappel NSW	f1 measure NSW
WNUT dev	baseline		0.3999	0.3542	0.3757
	(Leeman-Munk *et al.*, 2015)		0.9783	0.9930	0.9736
	google	rmsprop	0.7385	0.7965	0.7664
		adadelta	0.6952	0.8307	0.7569
		adagrad	0.7835	0.7954	0.7894
		sgd	0.7964	0.8188	0.8075
	interne	rmsprop	0.6023	0.8260	0.6966
		adadelta	0.7961	0.7821	0.7890
		adagrad	0.9157	0.8062	0.8574
		sgd	0.8931	0.7673	0.8254
WNUT dev+2577	baseline		0.3943	0.6459	0.4897
	google	rmsprop	0.6160	0.8606	0.7181
		adadelta	0.7000	0.8455	0.7659
		adagrad	0.8139	0.7893	0.8014
		sgd	0.7859	0.8184	0.8018
	interne	rmsprop	0.6254	0.8289	0.7129
		adadelta	0.8257	0.7849	0.8048
		adagrad	0.9038	0.7478	0.8145
		sgd	0.8769	0.7648	**0.8170**

TABLE 4 – Résultat d'évaluation

dev seul produit des modèles moins performants que WNUT dev+2577 tweets.

2. plongements : les plongements de Google sont moins efficaces que ceux développés en *interne* (nous avons aussi testé sans plongements pré-entrainés, le résultat n'est pas meilleur)
3. optimiser : le meilleur optimiser varie selon les expériences. En général, RMS prop donne le résultat le moins bon. Le sgd donne souvent un meilleur résultat.
   (nous avons testé aussi decay=learn rate/nb d'epochs, le résultat n'est pas meilleur)

6 Conclusion et perspectives du travail

Nous proposons dans cet article un modèle de réseau de neurones convolutif pour détecter dans des tweets les mots non-standards (NSW) à corriger. Nos expériences montrent que la quantité de données d'apprentissage influe sensiblement sur le résultat. Le modèle word2vec *interne* et l'optimiser SGD produisent le meilleur classifeur de NSW, avec une F1-mesure d'environ $0,83$ pour cette classe sur le corpus d'évaluation. Ce travail servira de base pour une étape de normalisation ultérieure des tweets.

Références

BALDWIN T., DE MARNEFFE M.-C., HAN B., KIM Y.-B., RITTER A. & XU W. (2015). Shared tasks of the 2015 workshop on noisy user-generated text : Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, p. 126–135, Beijing, China : Association for Computational Linguistics.

BEREND G. & TASNÁDI E. (2015). Uszeged : Correction type-sensitive normalization of english tweets using efficiently indexed n-gram statistics. In *Proceedings of the Workshop on Noisy User-generated Text*, p. 120–125, Beijing, China : Association for Computational Linguistics.

BERGSTRA J. & BENGIO Y. (2012). Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, **13**, 281–305.

COLLOBERT R., WESTON J., BOTTOU L., KARLEN M., KAVUKCUOGLU K. & KUKSA P. (2011). Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, **12**, 2493–2537.

DUCHI J., HAZAN E. & SINGER Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, **12**, 2121–2159.

GLOROT X. & BENGIO Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*.

HAN B. (2014). *Improving the utility of social media with Natural Language Processing*. PhD thesis, The University of Melbourne.

HAN B. & BALDWIN T. (2011). Lexical normalisation of short text messages : Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies - Volume 1*, HLT '11, p. 368–378, Stroudsburg, PA, USA : Association for Computational Linguistics.

HAN B., COOK P. & BALDWIN T. (2013). Lexical normalization for social media text. *ACM Trans. Intell. Syst. Technol.*, **4**(1), 5 :1–5 :27.

IOFFE S. & SZEGEDY C. (2015). Batch normalization : Accelerating deep network training by reducing internal covariate shift. In F. R. BACH & D. M. BLEI, Eds., *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, p. 448–456 : JMLR.org.

JIN N. (2015). Ncsu-sas-ning : Candidate generation and feature engineering for supervised lexical normalization. In *Proceedings of the Workshop on Noisy User-generated Text*, p. 87–92, Beijing, China : Association for Computational Linguistics.

KONG L., SCHNEIDER N., SWAYAMDIPTA S., BHATIA A., DYER C. & SMITH N. A. (2014). A dependency parser for tweets. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 1001–1012, Doha, Qatar : Association for Computational Linguistics.

LEEMAN-MUNK S., LESTER J. & COX J. (2015). Ncsu_sas_sam : Deep encoding and reconstruction for normalization of noisy text. In *Proceedings of the Workshop on Noisy User-generated Text*, p. 154–161, Beijing, China : Association for Computational Linguistics.

LI C. & LIU Y. (2014). Improving text normalization via unsupervised model and discriminative reranking. In *ACL (Student Research Workshop)*, p. 86–93.

LI C. & LIU Y. (2015). *Improving named entity recognition in tweets via detecting non-standard words*, In *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics*

and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference, volume 1, p. 929–938. Association for Computational Linguistics (ACL).

LIU F., WENG F. & JIANG X. (2012). A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics : Long Papers - Volume 1*, ACL '12, p. 1035–1044, Stroudsburg, PA, USA : Association for Computational Linguistics.

MA X. & HOVY E. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*.

MAUSAM, SCHMITZ M., BART R., SODERLAND S. & ETZIONI O. (2012). Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, p. 523–534, Stroudsburg, PA, USA : Association for Computational Linguistics.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013). Efficient estimation of word representations in vector space. *CoRR*, **abs/1301.3781**.

MILLER F. P., VANDOME A. F. & MCBREWSTER J. (2009). *Levenshtein Distance : Information Theory, Computer Science, String (Computer Science), String Metric, Damerau ?Levenshtein Distance, Spell Checker, Hamming Distance*. Alpha Press.

NGUYEN V. H., NGUYEN H. T. & SNASEL V. (2016). Text normalization for named entity recognition in vietnamese tweets. *Computational Social Networks*, **3**(1), 10.

PLANK B., HOVY D., MCDONALD R. T. & S GAARD A. (2014). Adapting taggers to twitter with not-so-distant supervision. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference : Technical Papers, August 23-29, 2014, Dublin, Ireland*, p. 1783–1792.

RITTER A., CLARK S., MAUSAM & ETZIONI O. (2011). Named entity recognition in tweets : An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, p. 1524–1534, Stroudsburg, PA, USA : Association for Computational Linguistics.

SUPRANOVICH D. & PATSEPNIA V. (2015). Ihs_rd : Lexical normalization for english tweets. In *Proceedings of the Workshop on Noisy User-generated Text*, p. 78–81, Beijing, China : Association for Computational Linguistics.

TIELEMAN T. & HINTON G. (2012). Lecture 6.5—RmsProp : Divide the gradient by a running average of its recent magnitude. COURSERA : Neural Networks for Machine Learning.

TOUTANOVA K. & MANNING C. D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora : Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, p. 63–70, Stroudsburg, PA, USA : Association for Computational Linguistics.

ZEILER M. D. (2012). ADADELTA : an adaptive learning rate method. *CoRR*, **abs/1212.5701**.