Enrichissement de corpus par approche générative et impact sur les modèles de reconnaissance d'entités nommées

Danrun Cao*,**, Nicolas Béchet*, Pierre-François Marteau*, Oussama Ahmia**

*Univ. Bretagne Sud, CNRS, IRISA, Rue Yves Mainguy, 56000 Vannes, France
prenom.nom@univ-ubs.fr
**OctopusMind, 2 Pl. Saint-Pierre, 44000 Nantes, France
p.nom@octopusmind.info

Résumé. Les applications industrielles de la reconnaissance d'entités nommées (REN) sont souvent confrontées à des corpus déséquilibrés. Ceci est en général nuisible à l'efficacité des modèles entraînés, notamment lorsqu'ils sont soumis à de nouvelles données. Dans cet article nous développons deux approches génératives pour enrichir des corpus dans le but d'améliorer la proportion des entités. Nous comparons l'impact de ces enrichissements sur des modèles de REN, en utilisant différents types de plongements lexicaux non-contextuels et contextuels exploités dans un modèle biLSTM-CRF en charge de l'extraction des entités. L'approche est évaluée sur une tâche de détection de reconduction de marché appliquée à un corpus constitué d'appels d'offres. Les résultats montrent d'une part que l'enrichissement proposé ne dégrade pas les résultats de détection sur le corpus initial et d'autre part améliore de manière significative les taux de détection sur un corpus n'ayant pas participé à l'apprentissage.

1 Introduction

Dans le domaine du Traitement automatique des langues (TAL), une entité nommée est définie comme étant un syntagme nominal se référant à un objet réel du monde. Cette entité peut représenter un objet concret ou abstrait, et sa nature peut varier en fonction du contexte d'application. La reconnaissance d'entités nommées (REN) consiste à identifier et à classer ces entités nommées à partir d'un texte non structuré.

L'exploitation de la REN dans un cadre industriel donne fréquemment lieu à la constitution de corpus déséquilibrés dans lesquels certaines entités peuvent être peu ou sous représentées, soit en raison de leur nature même, soit en raison du manque de données exploitables les concernant. Cette situation peut rendre la REN difficile, en perturbant la convergence des modèles lors des phases d'apprentissage. Il en résulte généralement un sur-apprentissage qui limite la capacité des modèles à généraliser correctement sur de nouvelles données.

Nous explorons dans ce travail l'intérêt des approches génératives pour l'enrichissement de données dans le but d'augmenter la proportion des entités minoritaires dans les données d'entraînement. Le reste de l'article est structuré de la manière suivante : nous introduisons le contexte de ce travail dans la section 2. La section 3 présente brièvement les travaux connexes

et décrit les méthodes d'enrichissement envisagées. La section 4 détaille le protocole expérimental mis en place et la section 5 présente et discute les résultats.

2 Contexte

La reconduction des marchés publics est une procédure qui permet à une entité publique, telle qu'une administration gouvernementale, de prolonger la durée d'un contrat existant avec un fournisseur ou un prestataire de services. Un marché peut être reconduit plusieurs fois, sous les mêmes conditions que le contrat initial ou non. Dans l'exemple de marché qui suit, la durée totale de reconduction est fixée à 3 ans, en incluant 3 périodes de 1 an.

> *Le présent marché prendra effet le 1 janvier 2024 pour une durée de 2 ans, reconductible 3 fois un an, soit jusqu'au 31 décembre 2028.*

Onze éléments d'informations peuvent servir à détecter l'échéancier de la reconduction, à savoir : la date de début, de fin et la durée du marché initial (`debutm`, `finm`, `delaim`), celles de la reconduction (`debutr`, `finr`, `delair`), de chaque période de reconduction (`debutp`, `finp`, `period`), le nombre de reconductions autorisées (`nb rec.`), et la durée totale du marché, toute période confondue (`dureem`). Certaines de ces informations sont indiquées de manière explicite dans les appels d'offres, mais d'autres le sont de manière implicite, car elles peuvent être déduites facilement à partir des autres informations disponibles.

Le travail présenté ici s'inscrit dans le cadre du développement d'un outil de détection de la reconduction pour l'entreprise OctopusMind [1]. La version actuelle de l'outil repose sur un modèle de champ aléatoire conditionnel (CRF, (Lafferty et al., 2001)). L'outil prend en entrée les paragraphes d'un appel d'offres contenant des informations sur la reconduction, et effectue une classification au niveau du *token* [2]. Chaque token est représenté par des caractéristiques qui le décrivent, ainsi que celles des 5 tokens précédents et suivants pour exploiter des caractéristiques *à longue portée* (Li et al., 2011). La partie gauche de la figure 1 illustre l'annotation du token `01/01/2016`. À la sortie du modèle CRF, un système à base de 15 règles interprète les annotations issues du CRF pour extraire les informations implicites lorsque cela est possible.

Le modèle CRF est entraîné avec validation croisée sur un corpus constitué de 2 707 marchés publiés entre 2015 et 2022 (nommé *Octo-2015/22* ci-après). Il compte 759 572 tokens dont 32 945 en entité. Certaines catégories sont très peu présentes, notamment `debutm` (31 tokens), `debutp` (171 tokens) et `finp` (125 tokens). Afin de réduire l'impact de ce déséquilibre lors de la validation croisée, nous avons effectué un découpage stratifié du corpus, afin que la proportion des entités de chaque *fold* respecte celle du corpus d'origine. Pour ce faire, nous transformons la tâche temporairement en une classification *multi-label* de texte, car l'ordre des étiquettes n'importe pas dans cette étape. Nous effectuons ensuite la stratification en fonction du nombre de tokens de chaque catégorie dans les phrases. Ce processus est implémenté grâce à `scikit-multilearn` (Szymański et Kajdanowicz, 2018). Les ratios des jeux de données d'entraînement, de validation et d'évaluation sont respectivement de 80%, 10% et 10%. Les résultats obtenus sont présentés dans la colonne `Octo-2015/22` du tableau 1.

Ensuite, nous avons collecté un nouveau corpus d'évaluation éloigné temporellement, afin d'évaluer si le modèle réussit à reconnaître aussi efficacement des dates et des contextes absents

1. https ://www.octopusmind.info/
2. *token* : la plus petite unité de texte, comme un mot, une lettre, ou un signe de ponctuation.

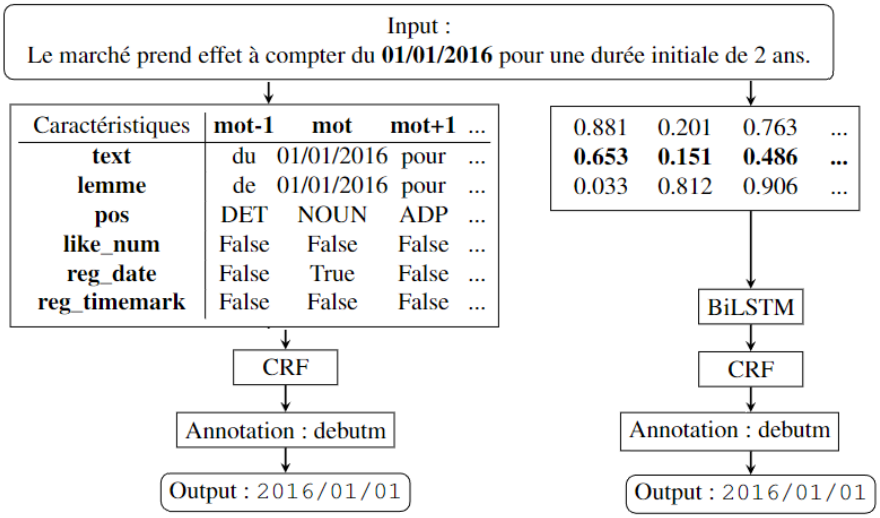

FIG. 1 – *Architecture du baseline CRF et les modèles biLSTM-CRF proposés.*

du corpus d'entraînement. Celui-ci est constitué de 350 annonces datant de 2011 (*Octo-2011* ci-après), et il compte 55 246 tokens dont 2 733 en entité. Nous évaluons les modèles de chaque *fold* sur ce corpus et nous calculons la moyenne comme le résultat final (cf tableau 1). À noter qu'il n'y a pas de date de `debutr` dans le corpus *Octo-2011*.

	Octo-2015/22			Octo-2011			perte		
	P	R	F1	P	R	F1	P	R	F1
debutm	93,6	93,0	93,3	88,5	50,1	63,7	-5,1	-42,9	-29,6
debutp	**86,9**	**66,2**	**71,9**	**100,0**	**27,8**	**43,3**	**13,1**	**-38,5**	**-28,7**
debutr	**40,0**	**31,1**	**34,0**	**-**	**-**	**-**	**-**	**-**	**-**
delaim	95,9	95,1	95,5	94,5	88,4	91,3	-1,5	-6,7	-4,2
delair	87,4	84,8	84,9	100,0	70,1	82,4	12,6	-14,7	-2,5
dureem	97,0	92,2	94,5	91,7	85,4	88,4	-5,3	-6,8	-6,1
finm	94,0	95,8	94,9	85,9	56,5	67,8	-8,2	-39,2	-27,0
finp	**78,0**	**65,6**	**69,9**	**87,8**	**18,8**	**30,7**	**9,8**	**-46,8**	**-39,2**
finr	91,0	82,2	85,7	74,9	69,7	71,7	-16,2	-12,5	-14,0
period	97,1	95,9	96,4	92,9	83,9	88,2	-4,2	-11,9	-8,3
nb rec.	98,7	98,1	98,4	99,1	94,6	96,8	0,3	-3,5	-1,6
macro m.	87,3	81,8	83,6	83,2	58,7	65,9	-4,1	-23,1	-17,7

TAB. 1 – *Résultats sur les corpus test et 2011.*

Lors de l'entraînement du modèle sur *Octo-2015/22*, nous constatons déjà un faible taux d'identification des catégories sous-représentées `debutr`, `period` et `finp`. De plus, lors de l'évaluation sur *Octo-2011*, nous remarquons une baisse générale de performance du mo-

dèle sur quasiment tous les champs d'évaluation, les dates représentant les catégories les plus impactées par cette baisse, notamment pour le rappel.

Outre le nombre limité d'exemples d'apprentissage, une autre justification possible à la baisse de performance est que le CRF développe un sur-apprentissage sur des dates spécifiques et ne peut donc pas généraliser. Dans un classement par importance des caractéristiques sur lesquelles le CRF s'appuie, nous constatons que pour la catégorie `debutm` par exemple, la caractéristique `text: 2023` occupe la 4ème place sur un ensemble de 2 942, et `text: 01/01/2023` la 9ème. Et pour `debutp`, `text: 1.6.2019` `text: 01/01/2024` se trouvent aux 19ème et 25ème places sur 795.

L'objectif de l'étude consiste à apporter des réponses aux deux problèmes évoqués (déséquilibre et sur-apprentissage) en exploitant des méthodes d'augmentation de données pour améliorer la proportion des entités dans le jeu de données, tout en limitant le risque de sur-apprentissage. Nous espérons qu'en enrichissant les données nous pouvons améliorer la robustesse du modèle, sans pour autant pénaliser sa performance sur le corpus d'entraînement.

Outre le déséquilibre des données, un autre facteur important qui conditionne l'efficacité d'un modèle est la représentation des données en entrée. Le modèle CRF actuel repose en effet sur les caractéristiques empiriques de chaque *token* ainsi que celles de son contexte immédiat. Si un contexte éloigné s'avère nécessaire, le modèle ne pourra pas fonctionner correctement. D'autre part, si nous augmentons la fenêtre de contexte, la taille de la matrice de features grossira rapidement et ceci réduira l'efficacité du modèle CRF. Afin de gérer l'information contextuelle d'une manière plus efficace, nous exploitons les méthodes de plongement lexical en vue d'obtenir une représentation plus *profonde* du texte, et pour finir, nous introduisons une couche biLSTM (Hochreiter et Schmidhuber, 1997) qui contrôlera la prise en compte du contexte de manière automatique. L'architecture de ces modèles REN est illustrée dans la partie droite de la figure 1. Les résultats de cette étude permettent d'obtenir une évaluation fine des méthodes de représentation et des modèles de reconnaissance exploités.

3 Enrichissement de données

Nous comparons deux stratégies d'enrichissement de données. La première repose sur des expressions régulières, inspirée par EDA (Wei et Zou, 2019), et la deuxième sur des modèles génératifs de langue. Dans le but d'améliorer la proportion des données, nous cherchons notamment à augmenter le nombre de *tokens* des catégories sous-représentées, i.e. `debutr`, `period` et `finp`, que nous appellerons les dates cibles ci-après.

La première approche `regex` consiste à remplacer dans le corpus les occurrences de dates par d'autres dates générées de manière aléatoire tout en restant plausibles. Pour cela, nous créons des expressions régulières permettant d'identifier les dates cibles. Ensuite, nous reproduisons cinq fois les textes contenant au moins une de ces dates cibles. Nous parcourons alors les dates dans chaque copie du texte et les remplaçons systématiquement par de nouvelles dates aléatoires. Nous prenons soin de diversifier les formats des dates générées et de simuler les erreurs que pourraient produire les créateurs d'annonces. Les données générées avec cette méthode demeurent en conformité avec la structure des appels d'offres réels. En contrepartie, la structure du texte autour des dates restant inchangée, cette méthode pourrait induire de fait un manque de variabilité dans les données générées, ainsi conduire à un sur-apprentissage du modèle sur des annonces relativement stéréotypées.

La deuxième approche consiste à utiliser un modèle génératif afin de compléter des *prompts* passés en entrée pour générer un paragraphe. Nous nous sommes orientés vers `davinci` (plus précisément `text-davinci-003` d'OpenAI). La première étape correspond à la préparation de *prompts*. Nous identifions en premier lieu des phrases contenant une date cible, puis sélectionnons les 20 mots la précédant. Ainsi, nous obtenons 132 *prompts* initiaux. Certains de ces prompts peuvent contenir des dates non ciblées, pouvant réduire la variabilité des paraphrases car le modèle se contente simplement de compléter le calendrier. Pour remédier à ce problème, nous créons 5 copies de chaque prompt, puis remplaçons la date par une nouvelle date générée aléatoirement. Cette étape d'augmentation produit un total de 386 prompts. `davinci` génère ensuite 10 paragraphes pour chaque *prompt*, soit 3 860 paragraphes attendus. Après l'élimination des doublons et des paragraphes non conformes, nous avons pu en conserver 1 159 (soit 30%) de très bonne qualité.

Type d'entité	debutm	finm	delaim	**debutr**	finr	delair
origine	3 142	2 811	10 367	**31**	666	298
regex	15 390	13 970	6 105	**150**	3 320	415
davinci	792	1 613	239	**355**	1 458	175
Type d'entité	period	nb rec.	**debutp**	**finp**	dureem	
origine	5 504	3 634	**171**	**125**	6 196	
regex	8 495	5 655	**745**	**625**	6105	
davinci	1 531	540	**2 030**	**1 999**	371	

TAB. 2 – *Nombre d'entités obtenu à partir des données générées (regex et davinci).*

Le décompte des données synthétisées par chaque méthode est présenté dans le tableau 2. En nombre absolu, la méthode à base d'expressions régulières fournit le plus de données synthétisées avec quasiment 500% de gain. Mais les données générées par `davinci` présentent une meilleure variabilité. Nous évaluons par la suite l'efficacité de l'introduction de ces données lors de l'apprentissage des modèles.

4 Protocole expérimental

5 modèles de plongement lexical sont comparés dans cette étude, à savoir : un fastText (Bojanowski et al., 2017) (300D) français pré-entraîné sur le corpus Common Crawls [3], un fastText (300D) "métier" pré-entraîné sur un corpus de marchés publics de l'entreprise OctopusMind, CamemBERT (Martin et al., 2020) (768D), BERT multilingue (Devlin et al., 2019) (768D) et XLM-R (Conneau et al., 2020) (768D). Ces choix ont été faits en accord avec les résultats publiés dans l'étude Cao et al. (2023), ces modèles étant en effet ceux qui ont conduit aux meilleures performances dans leur catégorie correspondante. Tous ces modèles sont finalement couplés à un modèle BiLSTM-CRF utilisé comme classifieur final. Cette procédure est implémentée à l'aide de la librairie `Flair` (Akbik et al., 2019) avec les mêmes configurations d'expérience de l'étude de Cao et al. (2023). Par ailleurs, nous comparons systématiquement un autre cas d'usage courant des modèles *transformer* : classifieur bout en bout (*end-to-end*).

3. https ://commoncrawl.org/

Nous reprenons la même procédure de validation croisée évoquée dans section 2, afin que les résultats soient comparables à ceux obtenus par notre *baseline* CRF. Les modèles sont entraînés une première fois sur le corpus d'origine sans ajout de données synthétisées, puis deux fois supplémentaires en exploitant les deux jeux synthétisés. Ces derniers sont ajoutés uniquement dans le jeu d'entraînement, et ne participent donc pas à la phase de validation d'entraînement, ni à l'évaluation finale. En plus de l'évaluation sur le corpus original, nous évaluons également les modèles sur *Octo-2011* pour vérifier leur robustesse.

Deux métriques d'évaluation sont proposées. Nous calculons d'abord le taux de documents pour lesquels toutes les annotations sont correctes. Ensuite, nous appliquons les règles de calcul (cf. supra) sur les annotations produites par les modèles, et nous évaluons une seconde fois le taux de documents correctement annotés. Cette dernière étape nous permet de connaître la performance d'un modèle lors de sa future exploitation pour l'entreprise.

5 Résultats

Les tableaux 3 et 4 présentent les résultats d'évaluation. L'ajout des données synthétisées permet, en général, d'améliorer la robustesse des modèles, car la performance sur *Octo-2011* augmente pour la plupart des modèles. Sur *Octo-2015/22*, l'introduction des données synthétisées peut entraîner des pertes légères sur le taux de réussite. Ceci s'explique en partie par le sur-apprentissage des modèles sur les entités sous-représentées dans *Octo-2015/22*, et le fait que celui-ci soit amélioré suite à l'augmentation de corpus. Nous constatons également que ces pertes ont été en partie corrigées par le système de règles. Sur *Octo-2015/22*, le meilleur taux de réussite est obtenu par XLM-R + BiLSTM-CRF avec les données `davinci`, avant et après l'application des règles. Sur *Octo-2011*, CamemBERT + biLSTM-CRF présente le meilleur résultat avant l'application des règles. Lorsque combiné avec les règles, c'est fastText métier + biLSTM-CRF avec les données `regex` qui obtient la meilleure performance. Cela signifie que si nous mettons ce modèle de reconnaissance dans la chaîne de traitement de l'entreprise, nous gagnerons 16,3% d'annonces correctement annotées, ce qui présente un gain considérable.

6 Conclusion

Nous avons présenté dans cet article deux méthodes d'augmentation de données en vue d'améliorer la porportion des entités nommées dans les corpus d'apprentissage. L'impact de cet enrichissement de données a été étudié, d'abord sur un modèle CRF actuellement mis en œuvre par l'entreprise, puis sur des classifieurs BiLSTM-CRF exploitant des plongements lexicaux. Pour les plongements à base de *transformers*, nous avons évalué leur performance avec et sans ajustement. Il ressort de notre étude que pour les modèles CRF et BiLSTM-CRF, l'introduction des données synthétisées est pertinente. Elle permet d'améliorer la généralisation des modèles sur de nouvelles données sans trop affecter leur performance sur le corpus initial.

Étant donné la nature payante du modèle `davinci`, nous avons dû limiter la quantité de données produite, et avons obtenu un corpus bien plus petit que celui produit par nos `regex`. Il aurait été intéressant d'harmoniser la taille des deux jeux d'enrichissement, afin de réduire ce biais dans nos résultats. Nous aurions pu également mieux combiner les méthodes d'augmentation, par exemple en introduisant à la fois les données générées par la méthode `regex` et

	avant règles					après règles				
	origine	regex	Δ	davinci	Δ	origine	regex	Δ	davinci	Δ
Descripteurs linguistiques										
baseline	*76,1*	78,8	2,7	76,5	0,4	*86,8*	88,2	1,4	86,7	-0,1
fastText										
cc	66,3	76,6	10,3	67,4	1,1	80,9	86,4	5,5	80,9	0
métier	67,1	70,5	3,4	68,3	1,2	80,2	81,7	1,5	80,5	0,3
CamemBERT										
biLSTM-CRF	80,6	75,1	-5,5	79,5	-1,1	88,9	85,1	-3,8	86,4	-2,5
bout en bout	76,1	74,5	-1,6	77,3	1,2	88,3	88,3	0	88,4	0,1
mBERT										
biLSTM-CRF	80,3	73,5	-6,8	79,7	-0,6	89,2	84,5	-4,7	88,7	-0,5
bout en bout	72,8	73,4	0,6	73,6	0,8	86,5	86,6	0,1	86,9	0,4
XLM-R										
biLSTM-CRF	66	68	2	**81,4**	15,4	79,2	81,6	2,4	**89,8**	10,6
bout en bout	73,3	73,4	0,1	74,4	1,1	87,4	86,8	-0,6	87,7	0,3

TAB. 3 – *Pourcentage de documents correctement annotés sur Octo-2015/22.*

	avant règles					après règles				
	origine	regex	Δ	davinci	Δ	origine	regex	Δ	davinci	Δ
Descripteurs linguistiques										
baseline	*56*	53,8	-2,2	57,9	1,9	*76,4*	75,1	-1,3	78,2	1,8
fastText										
cc	70,8	77,2	6,4	74,5	3,7	88,5	91,3	2,8	90,6	2,1
métier	80	80,5	0,5	79,2	-0,8	91,2	**92,7**	1,5	91,9	0,7
CamemBERT										
biLSTM-CRF	70,9	**83,9**	13	74,8	3,9	84,7	91,7	7	87,3	2,6
bout en bout	63,9	65,1	1,2	67,1	3,2	85,9	86	0,1	85,7	-0,2
mBERT										
biLSTM-CRF	78	82,3	4,3	75,6	-2,4	90,4	90,2	-0,2	90,7	0,3
bout en bout	62,8	64,6	1,8	63,1	0,3	84,1	83,2	-0,9	84,5	0,4
XLM-R										
biLSTM-CRF	76,7	77,1	0,4	71,5	-5,2	90	92	2	87,8	-2,2
bout en bout	63,1	62,6	-0,5	66,2	3,1	84,4	84,1	-0,3	87,7	3,3

TAB. 4 – *Pourcentage de documents correctement annotés sur Octo-2011.*

celles de `davinci`. Nous envisageons donc d'exploiter d'autres outils génératifs *open-source* comme par exemple LlaMa 2 [4]. Nous avons la possibilité de l'ajuster sur les données de l'entreprise, afin que le modèle produise des données appropriées à nos besoins.

Finalement, les analyses préliminaires des erreurs de reconnaissance produites par les différents modèles évalués permet d'envisager une prolongation de ce travail en proposant un méta-modèle susceptible d'exploiter plusieurs types de plongement.

4. https ://ai.meta.com/llama/

Références

Akbik, A., T. Bergmann, D. Blythe, K. Rasul, S. Schweter, et R. Vollgraf (2019). FLAIR: An Easy-to-Use Framework for State-of-the-Art NLP. In *NAACL 2019*, pp. 54–59.

Bojanowski, P., E. Grave, A. Joulin, et T. Mikolov (2017). Enriching Word Vectors with Subword Information. *TACL 5*, 135–146.

Cao, D., N. Béchet, et P.-F. Marteau (2023). Étude comparative des plongements lexicaux pour l'extraction d'entités nommées en français. In *18e CORIA – 16e RJC en RI – 30e TALN – 25e RECITAL*, Paris, France, pp. 11.

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, et V. Stoyanov (2020). Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th HLT-ACL*, pp. 8440–8451. ACL.

Devlin, J., M.-W. Chang, K. Lee, et K. Toutanova (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL 2019*, Minneapolis, Minnesota, pp. 4171–4186. ACL.

Hochreiter, S. et J. Schmidhuber (1997). Long Short-Term Memory. *Neural Computation 9*(8), 1735–1780.

Lafferty, J., A. Mccallum, et F. Pereira (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the ICML 2001*, ICML '01, San Francisco, CA, USA, pp. 282–289. Morgan Kaufmann Publishers Inc.

Li, Y., J. Jiang, H. L. Chieu, et K. M. A. Chai (2011). Extracting Relation Descriptors with Conditional Random Fields. In *Proceedings of IJCNLP '5*, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Martin, L., B. Muller, P. J. Ortiz Suárez, Y. Dupont, L. Romary, de la Clergerie, D. Seddah, et B. Sagot (2020). CamemBERT : a Tasty French Language Model. In *Proceedings of the 58th HLT-ACL*, pp. 7203–7219. Association for Computational Linguistics.

Szymański, P. et T. Kajdanowicz (2018). A scikit-based Python environment for performing multi-label classification. arXiv:1702.01460 [cs].

Wei, J. et K. Zou (2019). EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proceedings of EMNLP-IJCNLP 2019*, Hong Kong, China, pp. 6381–6387. Association for Computational Linguistics.

Summary

Industrial applications of Named Entity Recognition (NER) are usually confronted with imbalanced corpora. This could harm the performance of trained models when dealing with unknown data. In this paper we develop two generation-based data enrichment approaches to improve entity distribution. We compare the impact of enriched corpora on NER models, using both non-contextual and contextual embeddings, and a biLSTM-CRF as entity classifier. The approach is evaluated on a contract renewal detection task. The results show that the proposed enrichment significantly improves the model's effectiveness on unkonwn data, while not degrading the performance on the original test set.