Défi TextMine 2024 : "Reconnaissance d'entités géographiques dans un corpus des Instructions nautiques" - soumission équipe CRIT

Nicolas Gutehrlé

Défi TextMine 2024 : "Reconnaissance d'entités géographiques dans un corpus des Instructions nautiques" - soumission équipe CRIT

Nicolas Gutehrlé*

*Université de Franche-Comté, CRIT
F-25000 Besançon, France
nicolas.gutehrle@univ-fcomte.fr,
https://nicolasgutehrle.github.io/

Résumé. Cet article présente notre participation au défi TextMine 2024. Nous proposons une méthode reposant sur un modèle CRF linéaire, ainsi qu'une méthode hybride combinant un modèle CRF linéaire et un modèle XGBoost. Nous entraînons les modèles sur des caractéristiques morphologiques et syntaxiques extraites de chaque mot. Notre modèle CRF linéaire obtient une micro F-mesure de 0.942 sur le jeu de test privé. Ces résultats montrent que des modèles simples tels que CRF linéaire entraînés sur des caractéristiques simples telles que la forme des mots sont efficaces pour la tâche de reconnaissance d'entités spatiales nommées et non nommées.

1 Introduction

Le défi TextMine 2024 consiste en la reconnaissance d'entités spatiales nommées et non nommées à plusieurs niveaux dans les Instructions nautiques, une série d'ouvrages publiée par le Shom (Service hydrographique et océanographique de la Marine). Le corpus mis à disposition est constitué d'extraits de 15 volumes des Instructions nautiques décrivant des environnements maritimes côtiers. La tâche de reconnaissance des entités nommées (REN) a suscité un vif intérêt ces dernières années, et a vu l'application de nombreuses approches, en particulier les approches par règles (Nadeau et Sekine, 2007; Sekine et Nobata, 2004; Kim et Woodland, 2000; Farmakiotou et al., 2000; Appelt et al., 1993), par apprentissage automatique (*machine-learning*) (Asahara et Matsumoto, 2003; Borthwick et al., 1998; Sekine, 1998; Bikel et al., 1998; McCallum et Li, 2003; Ritter et al., 2011; Rocktäschel et al., 2012), et plus récemment par apprentissage profond (*deep-learning*) (Yadav et Bethard, 2019; Li et al., 2020)

Dans cet article, nous présentons notre participation au défi TextMine 2024. Nous abordons la reconnaissance d'entités spatiales nommées et non nommées comme une tâche d'annotation de séquence. Nous entraînons pour cela trois modèles CRF linéaires (Lafferty et al., 2001), chacun entraîné sur des ensembles de caractéristiques différents. De plus, nous proposons un modèle hybride, combinant un modèle CRF linéaire pour la détection des mentions d'entités nommées, et un modèle XGBoost (Chen et Guestrin, 2016) pour la classification des mentions détectées. Nous entraînons les modèles sur des caractéristiques (*features*) morphologiques et

syntaxiques extraites de chaque mot dans le jeu de données d'entraînement mis à disposition. Les résultats du défi sont disponibles sur Kaggle [1], tandis que notre code et nos expérimentations sont disponibles sur GitHub [2]. Le reste de l'article est structuré comme suit : nous présentons les données dans la section 2, puis notre méthodologie dans la section 3. Nous présentons l'évaluation de notre méthodologie en section 4 avant de présenter notre conclusion en section 5.

2 Données

Un jeu de données d'entraînement et de test ont été mis à disposition des participants, au format CSV. Le jeu d'entraînement est divisé en deux colonnes, une pour les tokens et une pour les classes correspondantes. Le jeu de test ne contient que les tokens. Le jeu de test mis à disposition, dit jeu de test public, représente environ 60 % des données totales de test. Afin d'entraîner nos modèles et identifier les meilleures combinaisons d'hyper-paramètres possibles, nous avons divisé le jeu d'entraînement en un jeu d'entraînement et un jeu de développement. Le Tableau 1 présente la distribution des classes dans les jeux de données d'entraînement et de développement.

Classe	Entraînement	Développement
aucun	22 859	9 804
geogFeat	1 899	805
geogName name	1 491	626
geogFeat geogName	995	468
geogName	655	255
Total	27 899	11 958

TAB. 1 – *Distribution des classes dans les jeux de données d'entraînement et de développement.*

Plusieurs classes sont composées de multiples étiquettes : par exemple, la classe *geogFeat geogName* est composée des étiquettes *geogFeat* et *geogName*, qui sont toutes deux des classes à part entière dans le jeu de données. Le Tableau 2 présente la distribution des classes uniques dans les jeux d'entraînement et de développement.

Classe	Entraînement	Développement
name	1 491	626
geogFeat	2 894	1 273
geogName	3 141	1 349

TAB. 2 – *Distribution des étiquettes uniques dans les jeux de données d'entraînement et de développement.*

1. https://www.kaggle.com/competitions/defi-textmine-2024/submissions
2. https://github.com/nicolasgutehrle/DefiTextmineCRIT

3 Méthodologie et implémentation

Nous abordons ce défi comme une tâche d'annotation de séquence. Afin d'entraîner les modèles, nous extrayons des caractéristiques morphologiques et syntaxiques pour chaque token t dans le jeu de données. Nous extrayons ces mêmes caractéristiques des token qui précédent et suivent directement le token t. Ces caractéristiques sont préfixées par *prev_* et *next_* respectivement. Le Tableau 3 présente l'ensemble des caractéristiques extraites pour un token t. La caractéristique $shape$ représente les lettres majuscules par un "X", les valeurs numériques par un "d" et les ponctuations par un ".". Tous les autres caractères sont représentés par un "x". Par exemple, l'entité nommée "Nadji" aura la forme "Xxxxx". Nous employons le modèle *fr_core_news_lg* du framework *spaCy* [3] pour obtenir les parties du discours ainsi que les rôles syntaxiques des tokens. Nous employons également la liste de mots vides de ce modèle pour déterminer si un token est un mot vide.

	Caractéristique	Description
1	$Token$	Mot
2	$lower$	Mot en minuscule
3	$isdigit$	Vrai si le mot est une valeur numérique, sinon Faux
4	$isupper$	Vrai si le mot est écrit en lettres capitales, sinon Faux
5	$ispunct$	Vrai si le mot est une ponctuation, sinon Faux
6	$isstop$	Vrai si le mot est un mot vide, sinon Faux
7	len	Nombre de caractères composant le mot
8	$shape$	Forme du mot
9	pos	Partie du discours du mot
10	dep	Rôle syntaxique du mot

TAB. 3 – *Caractéristiques extraites pour chaque token.*

Nous entraînons trois modèles CRF linéaires, chacun avec un ensemble différent de caractéristiques : $baseModel$ entraîné avec les caractéristiques 1 à 8, $posModel$, entraîné avec les caractéristiques 1 à 9 et $posDepModel$, entraîné avec les caractéristiques 1 à 10. Notre objectif est de comparer l'apport des parties du discours et des dépendances syntaxiques pour la reconnaissance d'entités spatiales nommées et non nommées. Nous utilisons l'implémentation du modèle CRF linéaire proposée par le package *sklearn-crfsuite* [4]. Pour chaque modèle, nous recherchons la combinaison optimale de valeurs d'hyper-paramètres parmi les valeurs présentées dans le Tableau 4. Nous évaluons chaque modèle sur le jeu de développement. Les valeurs optimales de chaque hyper-paramètre pour chaque modèle sont présentées dans le Tableau 5. Enfin, nous entraînons chaque modèle CRF linéaire sur l'ensemble des données, c'est-à-dire sur les jeux d'entraînement et de développement combinés, avec les meilleurs valeurs d'hyper-paramètres identifiées.

Comme indiqué en section 2, la classe d'un token peut être une combinaison de plusieurs étiquettes. Afin d'exploiter la nature composite des classes, nous proposons un modèle hy-

3. https://spacy.io/
4. https://sklearn-crfsuite.readthedocs.io/en/latest/

c1	0.5, 0.1, 0.01, 0.001, 0.0001
c2	0.5, 0.1, 0.01, 0.001, 0.0001

TAB. 4 – *Valeurs testées pour les hyper-paramètres c1 et c2 du modèle CRF linéaire.*

Modèle	c1	c2
$baseModel$	0.1	0.001
$posModel$	0.1	0.01
$posDepModel$	0.1	0.01

TAB. 5 – *Valeurs optimales des hyper-paramètres pour les modèles $baseModel$, $posModel$ et $posDepModel$.*

bride, qui combine un modèle CRF linéaire pour la détection d'entités nommées et un modèle XGBoost pour la classification de ces entités.

Afin d'entraîner le modèle CRF linéaire de ce modèle hybride, nous modifions les jeux d'entraînement et de développement de telle sorte que toutes les classes autre que "aucun" sont remplacées par l'étiquette "NER". Nous proposons ainsi une version binaire de jeu de données. La distribution des classes dans les jeux d'entraînement et de développement binaires est présentée dans le Tableau 6.

Classe	Entraînement	Développement
aucun	22 859	9 804
NER	5 040	2 154
Total	27 899	11 958

TAB. 6 – *Distribution des classes binaires dans les jeux de données d'entraînement et de développement.*

Nous entraînons plusieurs modèles CRF linéaires afin de trouver la combinaison optimale de valeurs d'hyper-paramètres parmi les valeurs présentées dans le Tableau 7. La valeur optimale de chaque hyper-paramètre y est notée en gras. Chaque modèle est évalué sur le jeu de développement. Nous entraînons enfin le modèle CRF linéaire binaire final sur l'ensemble des données avec les meilleurs valeurs d'hyper-paramètres identifiées.

Pour entraîner le modèle XGBoost, nous modifions les jeux de données d'entraînement et de développement de telle sorte qu'un token est associé à plusieurs classes. Ainsi, nous entraînons ce modèle selon une tâche classification multi-étiquettes (*multi-label*), et non pas une tâche de classification multi-classes. Nous employons l'implémentation du modèle XGBoost du package *xgboost*. Nous entraînons plusieurs modèles XGBoost afin de trouver la combinaison optimale de valeurs d'hyper-paramètres parmi les valeurs présentées dans le Tableau 8. La valeur optimale de chaque hyper-paramètre y est notée en gras. Chaque modèle est évalué sur le jeu de développement. Nous entraînons enfin le modèle XGBoost final sur l'ensemble des données avec les meilleurs valeurs d'hyper-paramètres identifiées.

c1	0.5, 0.1, 0.01, 0.001, **0.0001**
c2	0.5, **0.1**, 0.01, 0.001, 0.0001

TAB. 7 – *Valeurs testées pour les hyper-paramètres c1 et c2 du modèle CRF linéaire. Les valeurs en gras sont les valeurs optimales identifiées.*

n_estimators	50, 100, **150**, 200, 350, 500
max_depth	1, 2, 3, 4, 5, 6, 7, 8, **9**, 10
learning_rate	0.0001, 0.001, 0.01, 0.1, **1.0**
subsample	.25, .5, .75, **1**

TAB. 8 – *Valeurs testées pour les hyper-paramètres n_estimators, max_depth, learning_rate et subsample du modèle XGBoost. Les valeurs en gras sont les valeurs optimales identifiées.*

4 Résultats et Discussion

Les modèles ont été évalués en terme de micro F-mesure sur un jeu de test public et un jeu de test privé. Le jeu de test public représente environ 60 % du jeu de test privé. Le Tableau 9 présente les performances des quatre modèles sur les jeux de test public et privé.

Modèle	Micro F-mesure (public)	Micro F-mesure (privé)
$baseModel$	**0.964**	0.942
$posModel$	0.959	**0.944**
$posDepModel$	0.959	0.943
$hybrid$	0.915	0.895

TAB. 9 – *Evaluation des quatre modèles selon la micro F-mesure sur les jeux de test public et privé.*

Sur le jeu de test public, le modèle $baseModel$ atteint le meilleur score, qui est de 0.964. Les modèles $posModel$ et $posDepModel$ obtiennent tous deux un score de 0.959. Enfin, le modèle $hybrid$ obtient le score le plus bas, qui est de 0.915. Sur le jeu de test privé, le modèle $posModel$ atteint le meilleur score, qui est de 0.944. Les modèles $baseModel$ et $posDepModel$ obtiennent respectivement un score de 0.942 et 0.943. Enfin, le modèle $hybrid$ obtient le score le plus bas, qui est de 0.895.

Les scores des modèles $baseModel$, $posModel$ et $posDepModel$ sont similaires. L'ajout des parties du discours et des rôles syntaxiques comme caractéristiques d'entraînement ne semblent pas affecter les performances du modèle. De simples caractéristiques d'entraînements, telles que celles que nous avons extraites, semblent donc adéquates pour identifier les entités spatiales nommées et non nommées. Cependant, les données mises à disposition rassemblent les tokens dans un seul grand document, sans les séparer en phrases. Cela peut affecter la qualité de l'analyse en partie du discours ainsi que l'analyse en dépendances syntaxiques. Le modèle $hybrid$ obtient les scores les plus bas sur les deux évaluations. Cette baisse des scores peut s'expliquer par l'incapacité du modèle XGBoost à prendre en compte le

contexte de l'élément à catégoriser. La conversion de la tâche de classification multiclasses en une tâche de classification multi-étiquettes peut également expliquer ces résultats inférieurs.

Le Tableau 10 présente les dix plus importantes caractéristiques identifiées par le modèle $baseModel$ contribuant positivement à l'identification de chaque classe. Chaque caractéristique est représentée par sa valeur et son poids. La caractéristique *lower*, c'est-à-dire la forme du mot en minuscule, est la plus fréquemment employée pour chaque classe. Le modèle se concentre sur le mot lui-même pour les classes contenant l'étiquette *geogFeat*, comme en témoigne l'emploi des caractéristiques *Token* et *lower*. Il en est de même pour les classes contenant l'étiquette *geogName* et *name*. Pour ces classes là, le modèle se concentre également sur les mots précédents et suivants, comme en témoigne l'emploi des caractéristiques *prev_Token, prev_lower, next_Token, next_lower*. Le modèle semble apprendre du vocabulaire maritime ("mouillage", "rade", "lagon", etc.) pour identifier les classe *geogFeat* et *geogFeat geogName*. De même, il semble apprendre des noms propres désignant des lieux ("Islande", "Djibouti", etc.) pour identifier les classes *geogFeat geogName* et *geogName name*.

aucun	geogName	geogName name	geogFeat	geogFeat geogName
isdigit, 8.512	*Token* :», 6.085	*Token* :nouveau, 7.609	*lower* :mouillages, 11.987	*lower* :île, 10.255
lower :entre, 7.967	*lower* :», 6.085	*lower* :nouveau, 7.609	*lower* :amers, 9.929	*lower* :pointe, 8.422
Token :petit, 7.822	*Token* :principal, 4.995	*prev_lower* :côte, 7.492	*lower* :rade, 9.815	*lower* :ras, 7.632
lower :ouest, 7.780	*lower* :principal, 4.995	*lower* :djibouti, 7.405	*lower* :lagon, 9.778	*lower* :cap, 6.950
Token :grande, 6.443	*prev_lower* :capitainerie, 4.939	*lower* :blancpignon, 7.383	*lower* :alignement, 9.282	*lower* :îles, 6.755
lower :petite, 6.408	*next_Token* :médian, 4.922	*lower* :grande, 7.100	*Token* :appontement 9.074	*lower* :aéroport, 6.554
lower :jusqu', 5.881	*next_lower* :médian, 4.922	*lower* :espagnole, 6.454	*lower* :pontons, 9.057	*Token* :Island, 6.535
shape :X, 5.880	*shape* :xxxxxxxxx, 4.187	*Token* :ancien, 6.296	*Token* :baie, 8.515	*lower* :island, 6.535
lower :devant, 5.782	*prev_lower* :phare, 4.123	*lower* :ancien, 6.296	*lower* :échelles, 8.266	*Token* :Pass, 6.496
Token :Est, 5.746	*next_Token* :sur, 3.932	*shape* :Xxxxx.Xxxxxxx, 6.258	*lower* :îles, 8.249	*prev_Token* :ancienne, 6.251

TAB. 10 – *Dix plus importantes caractéristiques identifiées par le modèle $baseModel$ contribuant positivement à l'identification de chaque classe. Chaque caractéristique est représentée par sa valeur et son poids.*

5 Conclusion

Dans cet article, nous avons présenté notre participation au Défi TextMine 2024 : "Reconnaissance d'entités géographiques dans un corpus des Instructions nautiques". Nous avons proposé deux méthodes pour la tâche de reconnaissance d'entités spatiales nommées et non nommées. Notre première méthode repose sur un modèle CRF linéaire, tandis que notre seconde méthode est hybride, et combine un modèle CRF linéaire avec un modèle XGBoost. Notre modèle CRF linéaire, entraîné sur des caractéristiques morphologiques simples telles que la forme des mots, obtient une micro F-mesure de 0.964 sur le jeu de test public, et une micro F-mesure de 0.942 sur le jeu de test privé. Ces résultats montrent que des modèles simples tels que CRF linéaire entraînés sur des caractéristiques morphologiques également simples sont efficaces pour la tâche de reconnaissance d'entités spatiales nommées et non nommées. Dans des travaux futurs, nous comptons expérimenter avec d'autres caractéristiques d'entraînement d'ordre morphologique, syntaxique ou sémantique. Nous avons également l'intention d'approfondir notre étude sur la nature composite des entités nommées et non-nommées en continuant nos expérimentations sur la classification multi-étiquettes.

Références

Appelt, D. E., J. R. Hobbs, J. Bear, D. Israel, et M. Tyson (1993). Fastus : A finite-state processor for information extraction from real-world text. In *IJCAI*, Volume 93, pp. 1172–1178.

Asahara, M. et Y. Matsumoto (2003). Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the 2003 human language technology conference of the North American chapter of the association for computational linguistics*, pp. 8–15.

Bikel, D. M., S. Miller, R. Schwartz, et R. Weischedel (1998). Nymble : a high-performance learning name-finder. *arXiv preprint cmp-lg/9803003*.

Borthwick, A., J. Sterling, E. Agichtein, et R. Grishman (1998). Description of the mene named entity system as used in muc-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7), Fairfax, Virginia, April 29-May 1, 1998*.

Chen, T. et C. Guestrin (2016). Xgboost : A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.

Farmakiotou, D., V. Karkaletsis, J. Koutsias, G. Sigletos, C. D. Spyropoulos, et P. Stamatopoulos (2000). Rule-based named entity recognition for greek financial texts. In *Proceedings of the Workshop on Computational lexicography and Multimedia Dictionaries (COMLEX 2000)*, pp. 75–78.

Kim, J.-H. et P. C. Woodland (2000). A rule-based named entity recognition system for speech input. In *Sixth International Conference on Spoken Language Processing*.

Lafferty, J., A. McCallum, et F. C. Pereira (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data.

Li, J., A. Sun, J. Han, et C. Li (2020). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering 34*(1), 50–70.

McCallum, A. et W. Li (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons.

Nadeau, D. et S. Sekine (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes 30*(1), 3–26.

Ritter, A., S. Clark, O. Etzioni, et al. (2011). Named entity recognition in tweets : an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pp. 1524–1534.

Rocktäschel, T., M. Weidlich, et U. Leser (2012). Chemspot : a hybrid system for chemical named entity recognition. *Bioinformatics 28*(12), 1633–1640.

Sekine, S. (1998). Nyu : Description of the japanese ne system used for met-2. *http ://www. muc. saic. com/*.

Sekine, S. et C. Nobata (2004). Definition, dictionaries and tagger for extended named entity hierarchy. In *LREC*, pp. 1977–1980. Lisbon, Portugal.

Yadav, V. et S. Bethard (2019). A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv :1910.11470*.

Défi TextMine 2024 - équipe CRIT

Summary

This article presents our participation to the TextMine 2024 challenge. We propose a method based on a linear CRF model, as well as a hybrid method combining a linear CRF model and an XGBoost model. We train the models on morphological and syntactic features extracted from each word. Our linear CRF model obtains a micro F-measure of 0.942 on the private test set. These results show that simple models such as linear CRF trained on simple features such as word shape are effective for the Named and Unnamed Spatial Entity Recognition task.