Actes de CORIA-TALN 2023
Paris, France, 5 au 9 juin 2023

Étude comparative des plongements lexicaux pour l'extraction d'entités nommées en français

Danrun Cao[1, 2] Nicolas Béchet [1] Pierre-François Marteau [1]
(1) Univ. Bretagne Sud, CNRS, IRISA, Rue Yves Mainguy, 56000 Vannes, France
(2) OctopusMind, 2 Pl. Saint-Pierre, 44000 Nantes, France
danrun.cao@univ-ubs.fr, nicolas.bechet@univ-ubs.fr,
pierre-francois.marteau@univ-ubs.fr

RÉSUMÉ

Dans ce papier nous présentons une étude comparative des méthodes de plongements lexicaux pour le français sur la tâche de Reconnaissance d'entités nommées (REN). L'objectif est de comparer la performance de chaque méthode sur la même tâche et sous les mêmes conditions de travail. Nous utilisons comme corpus d'étude la proportion française du corpus WikiNER. Il s'agit d'un corpus de 3,5 millions tokens avec 4 types d'entités. 10 types de plongements lexicaux sont étudiés, y compris les plongements non-contextuels, des contextuels et éventuellement ceux à base de transformer. Pour chaque plongement, nous entraînons un BiLSTM-CRF comme classifieur. Pour les modèles à base de transformer, nous comparons également leur performance sous un autre cas d'usage : *fine-tuning*.

ABSTRACT

Comparative study of word embeddings for French Named Entity Recognition

In this paper we present a comparative study of word embedding methods for French named entity recognition (NER). Our objectif is to compare each method's performance when facing the same task and under the same working conditions. We use the French proportion of WikiNER as corpus of study. It is a 3.5-million token corpus with 4 entity types. 10 embedding methods are studied, including non-contextual ones, contextual ones and transformer-based ones. For each embedding, we train a BiLSTM-CRF as token classifier. for transformer-based models, we also compare their performance under another usage which is fine-tuning.

MOTS-CLÉS : Plongements lexicaux, Reconnaissance d'entités nommées, état de l'art.

KEYWORDS: Word embeddings, Names entity recognition, benchmark.

1 Introduction

En traitement automatique des langues (TAL), une entité nommée est définie comme étant un syntagme nominal se référant à un objet réel du monde. Ce syntagme est appelé la "mention" de l'entité en question. Une entité peut avoir plusieurs mentions. Par exemple "*France*" et "*République Française*" sont deux mentions renvoyant à l'entité "*France*". Ces objets peuvent être abstraits ou concrets, en fonction des besoins de chaque cas d'application. Il est important que la mention permette aux lecteurs d'identifier l'entité cible de manière précise et unique. Ceci sous entend qu'une mention ne peut être associée qu'à une seule entité. Quelques exemples courants d'entités nommées correspondent à des noms de personnes ou d'entreprises, des endroits, des dates, des évènements, etc.

Actes de la 30e Conférence sur le Traitement Automatique des Langues Naturelles (TALN), volume 1 : travaux de recherche originaux – articles longs, pages 94–104.

La tâche de reconnaissance d'entités nommées (REN) consiste à identifier ces entités dans un texte non structuré et à leur attribuer une liste de catégories prédéfinies. La tâche CoNLL 2003 (Sang & De Meulder, 2003) en est un exemple bien connu. Elle propose deux corpus annotés en anglais et en allemand. Quatre types d'entités sont traités : personne (PER, *person*), endroit (LOC, *location*), organisation (ORG, *organization*) et divers (MISC, *miscellaneous*). La REN est non seulement une tâche importante en soi, mais elle constitue aussi un composant clé pour des tâches plus complexes comme l'extraction de relations sémantiques, le résumé automatique, la fouille d'opinion, etc.

Historiquement, la tâche REN est abordée avec des méthodes à base de règles. Ces règles sont développées par des experts du domaine correspondant. Plusieurs pistes peuvent être suivies, par exemple en s'appuyant sur les caractéristiques morphologiques des entités (majuscule, minuscule, expressions régulières...), les informations lexicales et syntaxiques des mots, et éventuellement des ressources complémentaires (dictionnaires terminologiques, bases de connaissance...). Un des premiers travaux publiés de ce genre est (Rau, 1991), où l'auteure résout un problème de détection de noms d'entreprise par un système de règles symboliques. La méthode repose sur la mise en oeuvre de règles empiriques, et elle est peu coûteuse en temps de calcul. Cependant, elle est dépendante d'une intervention humaine importante et de ressources lexicales propres au domaine traité, ce qui constitue sa principale faiblesse. Ces ressources peuvent être coûteuses à produire. Il n'est pas toujours aisé d'y accéder, pour des raisons de propriétés intellectuelles et de moyen d'acquisition. Par ailleurs, les règles, ainsi que les ressources lexicales, nécessitent des mises à jour régulières, ce qui constitue un travail fastidieux.

Les approches plus récentes pour la REN privilégient les méthodes par apprentissage automatique. Elles prennent en entrée du texte brut, le vectorisent et utilisent un modèle de classification pour produire des annotations d'entité au niveau du token. On entend par "token" une séquence continue de caractères non vides, qui constitue l'unité minimale prise en compte lors du traitement automatique des données textuelles. Au début, ce vecteur de token se définit manuellement. Nous choisissons des caractéristiques et/ou des heuristiques permettant de capturer les spécificités d'un token, et leur valeur constitue alors le vecteur du token. Nous entraînons ensuite un annotateur automatique, tels que SVM et CRF, qui classifie les tokens en fonction de leur vecteur (Shen et al., 2004; Kazama & Torisawa, 2007).

Dans cette étude, nous nous intéressons à une méthode qui consiste à projeter les tokens dans un espace vectoriel continu, et ce de manière automatique. Cette méthode, que nous appellerons plongement lexical dans le reste de l'article, permettent d'encoder plus ou moins efficacement l'espace sémantique d'un token et éventuellement d'un texte entier. Cet article présente une étude comparative, pour la langue française, des méthodes de plongement issues de l'état de l'art sur une tâche de REN, en exploitant une même méthode de classification au niveau token. Cette étude permet d'analyser quantitativement l'impact des différents plongements lexicaux sur les résultats obtenus.

2 Travaux connexes

2.1 Plongement lexical

On entend par "plongement lexical" la représentation vectorielle des mots dans un espace métrique. Cet espace peut être interprété comme un espace sémantique de la langue, pour lequel chaque dimension correspond à un concept sémantique concret ou abstrait. La valeur des différentes dimensions décrit

alors la "proportion" de présence de chaque concept dans un mot, ce qui constitue une représentation quantifiée du "sens" du mot.

Nous distinguons deux types de plongements : non-contextuel et contextuel. Pour les plongements non-contextuels, chaque mot est associé à un vecteur unique, quel que soit le contexte dans lequel il est utilisé. Le premier algorithme de ce genre est *SENNA* (Collobert et al., 2011). *SENNA* initialise des vecteurs aléatoires pour chaque mot du vocabulaire et les entraîne de manière conjointe avec le classifieur en charge d'étiqueter les tokens. Il est possible d'ajouter des descripteurs empiriques comme ceux utilisés dans les systèmes à base de règles. Pour ce faire, *SENNA* génère des vecteurs aléatoires pour chacune des valeurs de ces descripteurs, et les concatène aux vecteurs de mots. Ces vecteurs de descripteurs sont également entraînés conjointement avec l'étiqueteur. Un peu plus tard, (Mikolov et al., 2013) ont proposé *Word2Vec*. Son concept clé est qu'un mot peut être défini par son contexte, et que les mots qui partagent un même contexte sont potentiellement des synonymes, lesquels pourront alors être représentés par des vecteurs proches. *fastText* (Bojanowski et al., 2017) est une extension de *Word2Vec* qui exploite les informations des sous-mots, c.-à-d. les n-grammes de caractères qui composent les mots. Cela permet notamment de prendre en compte les affixes et les racines des mots, et de mieux gérer les mots hors vocabulaire. Dans une autre approche, *GloVe* (Pennington et al., 2014) calcule les vecteurs en s'appuyant sur les co-occurrences des mots. Un dernier exemple de plongement non-contextuel est proposé dans (Chiu & Nichols, 2016). Ce modèle génère des vecteurs de mot par agrégation de vecteurs définis au niveau des caractères. L'ordre et la composition des caractères permettent également de représenter l'espace sémantique d'un mot. Chaque caractère initialement reçoit un vecteur aléatoire.

Les plongements non-contextuels ont montré de belles performances dans plusieurs tâches de TAL telles que l'analyse de sentiment, la fouille d'opinion, etc. Cependant ils ne traitent pas correctement les cas d'ambiguïté comme les polysèmes et les expressions figées. Pour corriger cette lacune, les plongements contextuels ont été introduits. Ils ont la particularité de travailler au niveau de la phrase plutôt qu'au niveau des mots isolés. *ELMo* (Peters et al., 2018) est un réseau de neurones constitué de deux couches de *Long Short-Term Memory (LSTM)* (Hochreiter & Schmidhuber, 1997) bidirectionnel (*BiLSTM*) qui traite une phrase vectorisée en entrée. *LSTM* est un réseau de neurones récurrents spécialisé dans le traitement des données séquentielles. En plus de sa capacité à "mémoriser" les entrées précédentes, il décide également à quel point cette mémoire doit être prise en compte dans l'étape courante. Un *biLSTM* est donc un *LSTM* qui considère la séquence de données dans les deux sens : vers l'avant (*forward*) et vers l'arrière (*backward*). Ce processus produit cinq représentations d'un mot : la suite de vecteurs de mots, 2 représentations *forward*, et 2 *backward*. Le vecteur final est produit par une combinaison pondérée de ces cinq représentations, les poids étant dépendants de la tâche à réaliser. Un autre exemple de plongement contextuel est *flair* (Akbik et al., 2018). *flair* considère une phrase comme une suite de caractères, et ce dans les deux sens. Un *LSTM* est entraîné à prédire le prochain caractère étant donné les précédents (ou les suivants). La représentation finale d'un mot est obtenue par concaténation de deux éléments : l'encodage du premier caractère de la phrase jusqu'au dernier caractère du mot, et celui du dernier caractère de la phrase jusqu'au début du mot.

Les architectures récentes à base de *Transformer* (Vaswani et al., 2017) se sont rapidement imposées dans l'état de l'art des méthodes de vectorisation. Une caractéristique importante du modèle Transformer est le mécanisme d'attention. Ce dernier permet au modèle de mesurer l'impact de chacun des autres tokens du texte lors du traitement du token courant, et ce en optimisant les poids d'attention. Le modèle tient compte également de l'ordre d'occurrence des tokens, en exploitant un vecteur qui encode l'emplacement de chaque token. Le premier modèle de plongement à base de *Transformer*

publié est *BERT* (Devlin et al., 2019). Il reprend la structure de Transformer d'origine, à savoir 12 couches identiques. Le modèle est entraîné sur deux tâches cibles non-supervisées : prédiction du token caché (*Masked Language Modeling*, *MLM*) et prédiction de la prochaine phrase (*Next Sentence Predictions*, *NSP*). Ces deux objectifs permettent au modèle de capturer à la fois les informations lexicales et phrastiques. Depuis, de nombreux variants de *BERT* ont été proposés, notamment :

— *RoBERTa* (Liu et al., 2019) et *DistilBERT* (Sanh et al., 2020) qui apportent des modification au niveau de la structure du réseau.

— *CamemBERT* (Martin et al., 2020), *FlauBERT* (Le et al., 2020) et *PhoBERT* (Nguyen & Tuan Nguyen, 2020) qui sont spécialisés sur d'autres langues que l'anglais.

À part *BERT* et les modèles dérivés, un autre modèle *Transformer*, *XLM* (Lample & Conneau, 2019), a été proposé. *XML* reprend également la structure de Transformer d'origine, mais en exploitant trois tâches d'entraînement : *MLM* comme dans *BERT*, prédiction du prochain token (*Causal Language Modeling*, *CLM*), et traduction (*Translation Language Modeling*, *TLM*). Un variant *XLM-R* (Conneau et al., 2020) basé sur l'architecture de *RoBERTa* a également été développé.

2.2 REN pour le français

En analysant les travaux effectués sur la REN en français, nous avons constaté un manque de corpus volumineux en français librement accessibles pour le développement des modèles de REN.

Le premier corpus REN en langue française librement accessible est proposé par (Sagot et al., 2012). Il s'agit de l'annotation d'entités nommées sur le corpus French Treebank (FTB) (Abeillé et al., 2003). Le corpus comprend 5 890 phrases issues du journal *Le Monde*, avec un total de 11 636 entités. Sur ce corpus, (Dupont, 2017) a entraîné et proposé l'outil SEM. La méthode s'appuie sur des descripteurs empiriques tels que les affixes, les noms communs précédents et suivants l'entité, ainsi que sur des lexiques. Un modèle à base de champ aléatoire conditionnel (*conditional random field* ou *CRF*, (Lafferty et al., 2001)) est ensuite entraîné puis utilisé comme classifieur. Ce système obtient un F1 score de 83,7% sur le corpus FTB-NER. (Suárez et al., 2020) ont proposé une comparaison entre SEM, *fastText*, *CamemBERT* et *FrELMo*. Le meilleur résultat est obtenu par la combinaison de *fastText* et *CamemBERT*, et le classifieur *LSTM-CRF*, le F1 score atteignant 90,25%. Les auteurs de *CamemBERT* ont également évalué leur modèle sur FTB-NER. Ils emploient *CamemBERT* en plongement et entraînent un LSTM-CRF. Le F1 score est de 89,55%.

Europeana Newspapers (Neudecker, 2016) est un autre corpus REN multilingue en accès libre, qui inclut le français, constitué d'articles de journaux numérisés avec des outils d'OCR. Cependant, l'OCR utilisé introduit de nombreuses erreurs et le corpus nécessite un travail de correction additionnel.

Enfin, FENEC (Millour et al., 2022) est un corpus contenant six genres de textes (prose, poésie, informations, encyclopédie, parole et multi-sources), annoté en entité nommée sous le schéma Quaero (Rosset et al., 2011). Ce corpus contient 11 149 tokens et 875 entités.

3 Corpus

Nous choisissons la proportion française du corpus WikiNER (Nothman et al., 2013) comme corpus d'étude, nous le désignerons WikiNER-fr dans la suite. WikiNER-fr comprend 134 092 phrases et 3.5 millions tokens, issus de plus de 61 000 articles Wikipédia. Il concerne les mêmes type d'entités que

ceux exploités dans CoNLL 2003, à savoir PER, LOC, ORG et MISC. Le tableau 1 montre le nombre de tokens pour chaque type d'entités dans le corpus.

Entité	PER	LOC	ORG	MISC
Nb. de tokens	129 978	155 565	45 443	81 594

TABLE 1 – Nb. de tokens pour chaque type d'entités dans le corpus

Les annotations de WikiNER ont été produites de manière semi-supervisée. D'abord les auteurs ont catégorisé manuellement environ 2 500 articles. Ensuite un classifieur est entraîné sur ce corpus initial puis exploité pour annoter le reste des pages. Parmi les trois modèles de classification comparées, le meilleur est la Régression logistique, avec un F1 score autour de 93% pour toutes les langues.

Puis la catégorie de chaque page Wikipédia est projetée sur les mentions associées aux hyperliens qui pointent vers la page en question. Par exemple, si la page "France" est annotée LOC, et si dans la phrase "Foucault retourne en France en 1960", le mot "France" reçoit un hyperlien vers la page "France", alors le mot "France" sera étiqueté LOC. Dans un article Wikipédia, seule la première occurrence d'une entité reçoit un hyperlien. Les auteurs ont alors mis en place quatre stratégies d'inférence afin de repérer les autres mentions des entités dans le corpus. 5 variants du corpus sont proposés, du corpus le plus bruité/exhaustif au corpus le moins bruité/exhaustif. Dans notre étude, nous avons choisi le variant wiki-2, un bon compromis produit en considérant l'avant-dernier niveau d'inférence (Nothman et al., 2013).

A noter que, mis à part le corpus d'entraînement initial, aucune vérification manuelle n'a été réalisée sur l'ensemble du corpus. Le corpus est alors dit en version *silver-standard*. Les auteurs ont montré que les modèles entraînés sur un corpus *silver-standard* pouvaient atteindre une performance comparable, ou parfois mieux dans certains cas, à ceux appris sur un corpus *gold-standard*, c.-à-d. sur un corpus révisé manuellement. Or cette comparaison a été faite uniquement sur le sous-corpus anglais. Dans le but de vérifier cette propriété pour le Français, nous avons choisi aléatoirement 20% des phrases du WikiNER-fr et révisé manuellement les annotations. Le processus de révision est détaillé dans (ref anonyme). Nous appellerons ce corpus révisé WikiNER-fr-gold dans la suite.

4 Expériences

4.1 Plongements lexicaux non-contextuels

Parmi les plongements lexicaux non-contextuels, nous avons évalué *Word2Vec*, *fastText* et *GloVe*. Pour *Word2Vec*, nous avons exploité le modèle pré-entraîné par Jean-Philippe Fauconnier. Ce modèle a été entraîné sur le corpus frWaC (Baroni et al., 2009), un ensemble de textes issus du WEB constitué de 1,6 milliards tokens. Pour *Glove*, nous avons entraîné un nouveau modèle sur le corpus frWac, afin que les résultats puissent être comparés à ceux obtenus par *Word2Vec*. Nous le mettrons à disposition de la communauté. Finalement pour fastText, nous utilisons le modèle proposé par l'équipe développeuse pré-entraîné sur le corpus Common Crawl. Nous étudions également la contribution des n-grammes de caractères de *fastText* pour la REN.

4.2 Plongments lexicaux contextuels

7 types de plongements lexicaux contextuels sont évalués, dont cinq exploitent l'architecture de type *transformer*. Les 2 modèles *non-transformer* sont *flair* et *ELMo*. Pour *flair*, 3 modèles sont évalués : un modèle français entraîné sur Wikipédia français, un autre modèle français entraîné sur frWaC, et un modèle multilingue entraîné sur le corpus JW300 (Agić & Vulić, 2019). Ce dernier est un corpus parallèle de plus de 300 langues, dont le français. Concernant *ELMo*, il existe un modèle français pré-entraîné (Che et al., 2018), disponible dans la librairie ELMoForManyLangs.

Parmi les 5 modèles transformers, nous distinguons là encore deux catégories : monolingue et multilingue. Les modèles français évalués sont *CamemBERT* et *DistilCamemBERT* (Delestre & Amar, 2022). Ce dernier est un travail inspiré par *DistilBERT* appliqué sur *CamemBERT*. Les trois autres modèles multilingues sont : *BERT* multilingue, *DistilBERT* multlingue, et *XLM-R*. Certains de ces modèles proposent des variants entraînés sur différents corpus ou avec des nombres différents de paramètres. Nous évaluons tous ces variants mais ne présentons que les résultats du meilleur variant dans la section 5. Il existe deux cas d'usage courants des modèles transformers. Le premier consiste à rajouter une couche de type perceptron comme classifieur, puis à poursuivre l'entraînement d'un modèle pré-entraîné sur la tâche cible. Ce processus est appelé *fine-tuning*. Cela permet au modèle d'optimiser ses paramètres spécifiquement pour la tâche cible. Une fois *fine-tuné*, le modèle pourra être exploité sans tenir compte de la couche de classification. Le deuxième correspond aux situations dans lesquelles un modèle pré-entraîné sert uniquement à produire des vecteurs de mot statiques. Ces vecteurs sont utilisés pour entraîner un classifieur sur les deux cas d'usage que nous comparons.

4.3 Procédure

Pour chaque plongement, nous entraînons un biLSTM-CRF et l'utilisons pour extraire et catégoriser les entités nommées en quatre classes. Cette procédure est implémentée à l'aide de la librairie Flair (Akbik et al., 2019). Le réseau biLSTM est initialisé avec 512 neurones (configuration par défaut de Flair). Le classifieur est entraîné sur un nombre d'époques inférieur à 150. Le corpus est découpé en trois sous ensembles : 60% entraînement, 20% validation et 20% test. Le jeu de test contient les mêmes phrases que WikiNER-fr-gold. Lors de l'entraînement, le modèle apprend sur les données d'entraînement et est évalué sur les données de validation. Une fois que le modèle a convergé, il est évalué d'abord sur les données de test puis sur WikiNER-fr-gold. Ceci nous permet de mesurer l'impact des erreurs du corpus sur la performance des modèles. Le taux d'apprentissage est initialisé à $0,1$. Celui-ci diminue en fonction de la qualité de l'entraînement obtenue à chaque époque.

A part le modèle fastText qui inclut les n-grammes, les plongements non-contextuels sont tous confrontés au problème des mots inconnus. Il est ainsi nécessaire de définir manuellement le vecteur à attribuer aux mots inconnus. Nous comparons ici deux stratégies pour proposer un vecteur "par défaut" pour les termes inconnus : (1) nous considérons la moyenne de tous les vecteurs du modèle ou (2) nous utilisons un vecteur nul, i.e. un vecteur dont toutes les dimensions prennent la valeur 0.

Certains des modèles à évaluer, comme ELMo et BERT, sont par nature des réseaux de neurones qui n'ont pas de couche de sortie permettant de produire des plongements directement. Ce sont les couches cachées qui encodent la langue. Il faut donc choisir une stratégie d'utilisation de ces couches cachées. Pour ELMo, nous concaténons l'état caché des 3 couches, c.-à-d. la couche d'entrée et les 2 couches de BiLSTM. Selon les auteurs (Peters et al., 2018), l'utilisation de la concaténation des couches permet une meilleure performance en général que d'une seule couche spécifique. Pour les

modèles à base de transformers, nous nous sommes référés aux résultats présentés dans l'article d'origine de BERT (Devlin et al., 2019). Les auteurs ont testé plusieurs stratégies sur le benchmark CoNLL 2003, et le meilleur résultat est obtenu avec la concaténation des états cachés des 4 dernières couches. Nous reprenons cette même configuration dans nos expériences.

L'évaluation finale est effectuée aux niveaux token et entité. Au niveau token, nous vérifions si la bonne catégorie d'entité a été attribué à un token. Au niveau entité, il faut que l'annotation et les frontières soient correctes. Nous utilisons la précision, le rappel et le F1 score comme métrique d'évaluation, et nous calculons la moyenne pondérée de ces métriques pour les quatre types d'entité.

5 Résultats

Le tableau 5 présente nos résultats. Le meilleur résultat pour chaque type de plongement est en gras, et le meilleur résultat sur l'ensemble des plongements est donné en bas du tableau en gras italique.

Nous remarquons en premier lieu une perte de performance générale quand nous évaluons les modèles sur WikiNER-fr-gold, avec une baisse de 1,11% en moyenne au niveau token et 1,85% au niveau entité. La seule exception est mBERT qui améliore légèrement sa performance. Les modèles étant entraînés sur le corpus *silver-standard*, cette perte est attendue car les modèles apprennent sur des données bruitées. Dans les annotations produites par le meilleur modèle de chaque catégorie, nous constatons les mêmes types d'erreurs et d'incohérences que ceux présents dans le corpus d'origine. Les modèles sont capables d'en corriger quelques unes, mais tout en produisant d'autres erreurs. Ceci confirme la nécessité d'une correction de corpus que nous avons décrit et initié dans cette étude.

Parmi les plongements non-contextuels, le meilleur résultat est obtenu par fastText avec les n-grammes. Il atteint un niveau de performance supérieur même à certains modèles à base de transformer. La deuxième place est occupée avec un petit écart par sa version sans n-grammes qui utilise un vecteur nul pour représenter les mots hors vocabulaire. L'ajout des n-grammes a exigé 5 fois plus de RAM (10,3Go avec n-grammes contre 2,05Go sans n-grammes), sans produire une amélioration considérable (+0.23% en moyenne). Concernant les deux stratégies de gestion de mots hors vocabulaire, le vecteur moyenne semble être une meilleure option pour Word2Vec (+1,03% en moyenne) et notamment GloVe (+8,3% en moyenne). En revanche, il semble préférable d'utiliser un vecteur nul pour les modèles fastText (+6,1% en moyenne).

Dans les deux cas d'usage des plongements à base de transformer, c'est CamemBERT en version large qui produit les meilleurs F1 scores, suivi par CamemBERT en version base et XLM-R en version large. Parmi les variants de CamemBERT-base, celui pré-entraîné sur Wikipédia obtient le meilleur résultat en mode statique, et celui sur CCNet (Wenzek et al., 2020) semble meilleur en mode *fine-tuning*. Toutefois, nous remarquons que l'écart entre CamemBERT-large et CamemBERT-base n'est pas très grand (0.5% en statique et 0.1% en *fine-tuning*). Sachant que la taille de CamemBERT-large est trois fois celle de CamemBERT-base (350m paramètres contre 110m), l'entraînement ainsi que l'exploitation du premier exigent bien plus de ressources de calcul que le second. Ceci pourrait être un facteur important à prendre en compte pour les cas d'usage industriels par exemple où on voudrait limiter le coût de ressources. Dans le même esprit, nous avons comparé mBERT avec DistilBERT-m. Malgré un nombre de paramètres plus faible (-40% d'après les auteurs), DistilBERT-m obtient quasiment la même performance que mBERT.

Plongement	WikiNER-fr test		WikiNER-fr-gold	
	F1 token	F1 entité	F1 token	F1 entité
Plongements non-contextuels				
Word2Vec+moyenne	89,6%	87,4%	88,2%	84,2%
GloVe+moyenne	89,3%	87,3%	87,9%	83,9%
fastText+moyenne	87,8%	84,8%	86,7%	82%
Word2Vec+nul	87,8%	86,1%	86,9%	84,5%
GloVe+nul	81,9%	76,9%	80,9%	75,5%
fastText+nul	92,7%	91,3%	91,7%	89,8%
fastText+n-grammes	**92,9%**	**91,6%**	**91,9%**	**90%**
Plongements contextuels non-transformer				
ELMo	90,6%	88,8%	89,7%	87,3%
flair français Wikipédia	91,0%	89,1%	90,1%	87,7%
flair français frwac	**91,9%**	**90,2%**	**90,6%**	**88,3%**
flair multilingue	89,7%	87,5%	88,7%	86,2%
Plongements transformer français statiques				
Camembert base ccnet	93,9%	92,6%	92,5%	90,5%
Camembert base wikipédia	94,3%	93,0%	93.2%	91,2%
Camembert large	**95%**	**93,7%**	**93,5%**	**91,4%**
DistilCamembert base	92,7%	91,1%	91,4%	89,2%
Plongements transformer multilingues statiques				
mBERT	93,0%	91,5%	91,7%	89,6%
DistilBERT-m	92,7%	91,2%	91,5%	89,4%
XLM-R base	91,1%	89,3%	89,9%	87,6%
XLM-R large	**93,9%**	**92,5%**	**92,5%**	**90,5%**
Plongements transformer français + fine-tuning				
Camembert base ccnet	94,0%	92,3%	92,8%	90,3%
Camembert large	**94,1%**	**92,5%**	**92,8%**	**90,5%**
DistilCamembert base	93,1%	91,0%	92%	89,3%
Plongements transformer multilingues + fine-tuning				
mBERT	92,0%	89,6%	92,4%	89,7%
DistilBERT-m	92,5%	90,1%	91,3%	88,2%
XLM-R base	92,5%	90,5%	91,3%	88,5%
XLM-R large	**93,6%**	**91,6%**	**92,6%**	**90%**
Meilleur résultat				
Camembert large en statique	***95%***	***93,7%***	**93,5%**	**91,4%**

TABLE 2 – Résultats d'évaluation des plongements

6 Conclusion

Nous avons évalué 10 types de plongements lexicaux sur le corpus WikiNER-fr. L'objectif est de montrer, face à une même tâche et dans les mêmes conditions initiales, quel modèle est le plus adapté à une tâche REN. Nous avons effectué une évaluation supplémentaire des modèles sur WikiNER-fr-gold, la version révisée d'une partie du corpus WikiNER-fr. Dans les deux cas, les modèles à base de transformer ont obtenu les meilleurs résultats. Nous constatons une perte de performance générale lors de l'évaluation sur WikiNER-fr-gold. Les erreurs constatées dans le corpus WikiNER-fr sont reproduites dans les annotations réalisées par les modèles entraînés sur ce corpus. Il est donc nécessaire de réviser le corpus. Enfin, il convient de noter qu'un nombre de paramètres important ne garantit pas un gain de performance signifiant.

Références

ABEILLÉ A., CLÉMENT L. & TOUSSENEL F. (2003). Building a Treebank for French. In A. ABEILLÉ, Éd., Treebanks : Building and Using Parsed Corpora, volume 20, p. 165–187. Dordrecht : Springer Netherlands. DOI : 10.1007/978-94-010-0201-1_10.

AGIĆ & VULIĆ I. (2019). JW300 : A Wide-Coverage Parallel Corpus for Low-Resource Languages. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, p. 3204–3210, Florence, Italy : Association for Computational Linguistics. DOI : 10.18653/v1/P19-1310.

AKBIK A., BERGMANN T., BLYTHE D., RASUL K., SCHWETER S. & VOLLGRAF R. (2019). FLAIR : An Easy-to-Use Framework for State-of-the-Art NLP. In NAACL 2019, p. 54–59.

AKBIK A., BLYTHE D. & VOLLGRAF R. (2018). Contextual String Embeddings for Sequence Labeling. In Proceedings of the 27th International Conference on Computational Linguistics, p.1̃2, Santa Fe, New Mexico, USA : Association for Computational Linguistics.

BARONI M., BERNARDINI S., FERRARESI A. & ZANCHETTA E. (2009). The WaCky wide web : a collection of very large linguistically processed web-crawled corpora. Language Resources and Evaluation, **43**(3), 209–226. DOI : 10.1007/s10579-009-9081-4.

BOJANOWSKI P., GRAVE E., JOULIN A. & MIKOLOV T. (2017). Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics, **5**, 135–146. DOI : 10.1162/tacl_a_00051.

CHE W., LIU Y., WANG Y., ZHENG B. & LIU T. (2018). Towards Better UD Parsing : Deep Contextualized Word Embeddings, Ensemble, and Treebank Concatenation. In Proceedings of the CoNLL 2018 Shared Task : Multilingual Parsing from Raw Text to Universal Dependencies, p. 55–64, Brussels, Belgium : Association for Computational Linguistics. DOI : 10.18653/v1/K18-2005.

CHIU J. P. C. & NICHOLS E. (2016). Named Entity Recognition with Bidirectional LSTM-CNNs. Transactions of the Association for Computational Linguistics, **4**, 357–370. DOI : 10.1162/tacl_a_00104.

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. DOI : 10.5555/1953048.2078186.

CONNEAU A., KHANDELWAL K., GOYAL N., CHAUDHARY V., WENZEK G., GUZMÁN F., GRAVE E., OTT M., ZETTLEMOYER L. & STOYANOV V. (2020). Unsupervised Cross-lingual Representation Learning at Scale. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, p. 8440–8451 : Association for Computational Linguistics. DOI : 10.18653/v1/2020.acl-main.747.

DELESTRE C. & AMAR A. (2022). DistilCamemBERT : a distillation of the French model CamemBERT. In CAp (Conférence sur l'Apprentissage automatique) : arXiv. DOI : 10.48550/ARXIV.2205.11111.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2019). BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, p. 4171–4186, Minneapolis, Minnesota : ACL. DOI : 10.18653/v1/N19-1423.

DUPONT Y. (2017). Exploration de traits pour la reconnaissance d'entités nommées du Français par apprentissage automatique. In Actes des 24ème Conférence sur le Traitement Automatique des Langues Naturelles. 19es REncontres jeunes Chercheurs en Informatique pour le TAL (RECITAL 2017), p. 42–55, Orléans, France : ATALA.

HOCHREITER S. & SCHMIDHUBER J. (1997). Long Short-Term Memory. Neural Computation, **9**(8), 1735–1780. DOI : 10.1162/neco.1997.9.8.1735.

KAZAMA J. & TORISAWA K. (2007). Exploiting Wikipedia as External Knowledge for Named Entity Recognition. In Proceedings of the 2007 EMNLP-CoNLL, p. 698–707, Prague, Czech Republic : Association for Computational Linguistics.

LAFFERTY J., MCCALLUM A. & PEREIRA F. (2001). Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01, p. 282–289, San Francisco, CA, USA : Morgan Kaufmann Publishers Inc. DOI : 10.5555/645530.655813.

LAMPLE G. & CONNEAU A. (2019). Cross-lingual Language Model Pretraining. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, p.Ĩ1, Red Hook, NY, USA : Curran Associates Inc. DOI : 10.5555/3454287.3454921.

LE H., VIAL L., FREJ J., SEGONNE V., COAVOUX M., LECOUTEUX B., ALLAUZEN A., CRABBÉ B., BESACIER L. & SCHWAB D. (2020). FlauBERT : Unsupervised Language Model Pre-training for French. In Proceedings of the Twelfth Language Resources and Evaluation Conference, p. 2479–2490, Marseille, France : European Language Resources Association.

LIU Y., OTT M., GOYAL N., DU J., JOSHI M., CHEN D., LEVY O., LEWIS M., ZETTLEMOYER L. & STOYANOV V. (2019). RoBERTa : A Robustly Optimized BERT Pretraining Approach. In Proceedings of the 20th Chinese National Conference on Computational Linguistics, p. 1218–1227, Huhhot, China : Chinese Information Processing Society of China.

MARTIN L., MULLER B., ORTIZ SUÁREZ P. J., DUPONT Y., ROMARY L., DE LA CLERGERIE , SEDDAH D. & SAGOT B. (2020). CamemBERT : a Tasty French Language Model. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, p. 7203–7219 : Association for Computational Linguistics. DOI : 10.18653/v1/2020.acl-main.645.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013). Efficient Estimation of Word Representations in Vector Space. DOI : 10.48550/ARXIV.1301.3781.

MILLOUR A., DUPONT Y., JOUGLAR A. & FORT K. (2022). FENEC : un corpus à échantillons équilibrés pour l'évaluation des entités nommées en français. In Actes de la 29e Conférence sur

le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale, p. 82–94, Avignon, France : ATALA.

NEUDECKER C. (2016). An Open Corpus for Named Entity Recognition in Historic Newspapers. In Proceedings of the Tenth International Conference on Language Resources and Evaluation, p. 4348–4352, Portorož, Slovenia : European Language Resources Association.

NGUYEN D. Q. & TUAN NGUYEN A. (2020). PhoBERT : Pre-trained language models for Vietnamese. In Findings of the Association for Computational Linguistics : EMNLP 2020, p. 1037–1042 : Association for Computational Linguistics. DOI : 10.18653/v1/2020.findings-emnlp.92.

NOTHMAN J., RINGLAND N., RADFORD W., MURPHY T. & CURRAN J. R. (2013). Learning multilingual named entity recognition from Wikipedia. Artificial Intelligence, **194**, 151–175. DOI : 10.1016/j.artint.2012.03.006.

PENNINGTON J., SOCHER R. & MANNING C. (2014). Glove : Global Vectors for Word Representation. In Proceedings of EMNLP 2014, p. 1532–1543, Doha, Qatar : Association for Computational Linguistics. DOI : 10.3115/v1/D14-1162.

PETERS M. E., NEUMANN M., IYYER M., GARDNER M., CLARK C., LEE K. & ZETTLEMOYER L. (2018). Deep contextualized word representations. In Proceedings of the 2018 NAACL : Human Language Technologies, Volume 1 (Long Papers), volume 1, p. 2227–2237, New Orleans, Louisiana : Association for Computational Linguistics. DOI : 10.18653/v1/N18-1202.

RAU L. (1991). Extracting company names from text. In [1991] Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application, volume i, p. 29–32, Miami Beach, FL, USA : IEEE Comput. Soc. Press. DOI : 10.1109/CAIA.1991.120841.

ROSSET S., GROUIN C. & ZWEIGENBAUM P. (2011). Entités nommées structurées : guide d'annotation Quaero.

SAGOT B., RICHARD M. & STERN R. (2012). Annotation référentielle du Corpus Arboré de Paris 7 en entités nommées. In Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, volume 2, p. 535–542, Grenoble, France : ATALA/AFCP.

SANG E. F. T. K. & DE MEULDER F. (2003). Introduction to the CoNLL-2003 Shared Task : Language-Independent Named Entity Recognition. In Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, p. 142–147.

SANH V., DEBUT L., CHAUMOND J. & WOLF T. (2020). DistilBERT, a distilled version of BERT : smaller, faster, cheaper and lighter. DOI : 10.48550/ARXIV.1910.01108.

SHEN D., ZHANG J., SU J., ZHOU G. & TAN C.-L. (2004). Multi-criteria-based active learning for named entity recognition. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, p. 589–es, Barcelona, Spain : Association for Computational Linguistics. DOI : 10.3115/1218955.1219030.

SUÁREZ P. J. O., DUPONT Y., MULLER B., ROMARY L. & SAGOT B. (2020). Establishing a New State-of-the-Art for French Named Entity Recognition. In Proceedings of the 12th LREC, p. 4631–4638, Marseille, France : European Language Resources Association.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. & POLOSUKHIN I. (2017). Attention Is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, p. 6000–6010, Long Beach, California, USA : Curran Associates Inc. DOI : 10.5555/3295222.3295349.

WENZEK G., LACHAUX M.-A., CONNEAU A., CHAUDHARY V., GUZMÁN F., JOULIN A. & GRAVE E. (2020). CCNet : Extracting High Quality Monolingual Datasets from Web Crawl Data. In Proceedings of the 12th LREC, p. 4003–4012 : European Language Resources Association.