Actes de la 29e Conférence sur le Traitement Automatique des Langues Naturelles
Avignon, France, 27 juin au 1er juillet 2022

Mieux utiliser BERT pour la détection d'évènements à partir de peu d'exemples

Aboubacar Tuo Romaric Besançon Olivier Ferret Julien Tourille
Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
{aboubacar.tuo,romaric.besancon,olivier.ferret,julien.tourille}@cea.fr

RÉSUMÉ

Les méthodes actuelles pour la détection d'évènements, qui s'appuient essentiellement sur l'apprentissage supervisé profond, s'avèrent très coûteuses en données annotées. Parmi les approches pour l'apprentissage à partir de peu de données, nous exploitons dans cet article le méta-apprentissage et l'utilisation de l'encodeur BERT pour cette tâche. Plus particulièrement, nous explorons plusieurs stratégies pour mieux exploiter les informations présentes dans les différentes couches d'un modèle BERT pré-entraîné et montrons que ces stratégies simples permettent de dépasser les résultats de l'état de l'art pour cette tâche en anglais.

ABSTRACT

Better exploitation of BERT for few-shot event detection

Recent approaches for event detection rely on deep supervised learning, which requires large manually annotated corpora. In the new approaches that are developed for few shot learning, we focus on meta-learning with a BERT encoder for the task. Specifically, we aim at optimizing the use of the information contained in the different layers of a pretrained BERT and show that simple strategies can be efficient and outperform the current state of the art for this task in English.

MOTS-CLÉS : Extraction d'évènements, apprentissage à partir de peu d'exemples, méta-apprentissage, BERT.

KEYWORDS: Event extraction, few-shot learning, meta-learning, BERT.

1 Introduction

L'extraction d'évènements consiste à extraire automatiquement des informations structurées concernant des événements à partir de textes. Elle peut être assimilée au remplissage d'un formulaire par des entités du texte, les champs de ce formulaire correspondant aux arguments de l'évènement. Si les premières méthodes d'extraction d'évènements s'appuyaient sur des règles élaborées manuellement (Ahn, 2006), elles ont peu à peu laissé la place à des techniques par apprentissage avec l'essor de l'apprentissage statistique et des réseaux neuronaux ces dernières années. Ainsi, Li *et al.* (2013) proposent un modèle de prédiction structurée fondé sur de nombreux traits lexico-syntaxiques, Nguyen & Grishman (2015) utilisent des réseaux convolutifs afin d'exploiter les informations contextuelles, Nguyen *et al.* (2016) proposent des modèles fondés sur les réseaux récurrents et Liu *et al.* (2018b), Nguyen & Grishman (2018) et Yan *et al.* (2019) utilisent des modèles de convolution de graphes pour capturer les dépendances syntaxiques entre les différentes parties d'une phrase.

Volume 1 : conférence principale, pages 392–402.

Bien que l'objectif de l'extraction d'évènements soit d'identifier tous les arguments, les jeux de données depuis ACE 2005 (Walker *et al.*, 2006) ont introduit le concept de déclencheur évènementiel (*trigger*), désignant le groupe de mots indiquant le plus explicitement possible la présence d'un évènement dans une phrase. Le but est de créer un ancrage lexical afin d'aider à la recherche des arguments par la suite. Nous nous focalisons dans ce travail sur la détection de ces déclencheurs évènementiels (*Event Detection ou Trigger Detection*).

Détection d'évènements à partir de peu d'exemples. Les méthodes par apprentissage supervisé sont très coûteuses puisqu'elles nécessitent de grands corpus annotés manuellement. Un des défis actuels est donc de développer des méthodes permettant de réduire, dans la mesure du possible, le coût de développement de ces systèmes. C'est dans ce contexte que nous envisageons la détection d'évènements par le biais de l'apprentissage à partir de peu d'exemples (*few-shot learning, FSL*) (Lake *et al.*, 2015).

La détection d'évènements à partir de peu d'exemples a fait l'objet de nombreuses études récentes. Diverses configurations ont été explorées : la généralisation de modèles à de nouveaux types d'évènements à l'aide de listes de mots clés (Bronstein *et al.*, 2015; Lai & Nguyen, 2019), l'enrichissement des données avec des ressources externes (Deng *et al.*, 2021), le zero-shot learning avec l'utilisation de description des classes ou l'utilisation de ressources externes (Zhang *et al.*, 2021) et le few-shot learning qui nous intéresse dans cette étude (Shen *et al.*, 2021; Chen *et al.*, 2021; Cong *et al.*, 2021). Certaines études actuelles se sont par ailleurs intéressées à la classification d'évènements à partir de peu d'exemples, laquelle consiste à attribuer un type d'événement à un candidat déclencheur événementiel déjà identifié dans une phrase (Lai *et al.*, 2021, 2020; Deng *et al.*, 2020).

Dans cet article, notre contribution se focalise sur une meilleure exploitation des représentations fournies par le modèle de langue BERT pour la détection d'évènements à partir de peu d'exemples, plus spécifiquement en étudiant l'importance de ces différentes représentations et en évaluant différentes façons de les associer.

2 Méthode proposée

2.1 Formulation du problème et notations

Nous formulons le problème de la détection d'évènements avec peu d'exemples comme un problème d'annotation de séquences (Ramshaw & Marcus, 1995), au format BIO (*Beginning Inside Outside*), qui peut être ramené à une tâche de classification multi-classe au niveau des tokens.

Les récentes études en FSL se rattachent au méta-apprentissage *(meta-learning*), souvent défini comme le fait d'apprendre à apprendre. L'idée principale du méta-apprentissage est d'entraîner des modèles avec un grand nombre de tâches diverses, chacune exploitant un nombre restreint d'exemples, afin que le modèle appris puisse rapidement reproduire ces tâches sur de nouvelles données. Les méthodes ayant émergé ces dernières années pour résoudre des tâches de FSL grâce au méta-apprentissage se regroupent en trois grandes catégories : des méthodes construisant des modèles spécifiques à la tâche (Yan *et al.*, 2015; Santoro *et al.*, 2016), d'autres fondées sur l'amélioration de l'algorithme d'optimisation (Finn *et al.*, 2017; Nichol *et al.*, 2018; Ravi & Larochelle, 2017) et enfin des méthodes fondées sur l'apprentissage d'une métrique (Vinyals *et al.*, 2016; Sung *et al.*, 2018; Snell *et al.*, 2017), incluant les réseaux prototypiques (Snell *et al.*, 2017) que nous utilisons dans ce travail.

Nous adoptons dans ce travail la formulation épisodique N ways, k shots comme décrite dans Vinyals *et al.* (2016). À chaque épisode, le modèle prend en compte un sous-ensemble de données étiquetées $\mathcal{S}$, appelé *support set*, qui contient N types d'évènements, et k exemples annotés par type (k étant généralement petit, par exemple égal à 1, 5 ou 10) :

$$\mathcal{S} = \{(x_1^1, t_1^1, y^1), \ldots, (x_k^1, t_k^1, y^1), \ldots, (x_1^N, t_1^N, y^N), \ldots, (x_k^N, t_k^N, y^N)\}$$

où $x_i^n = \{w_1, \ldots, w_L\}$ est une séquence de longueur L contenant un déclencheur événementiel de la classe n ; t_i^n la position du déclencheur et y^n la séquence d'étiquettes associée. Nous disposons par ailleurs d'un autre sous-ensemble similaire $\mathcal{Q}$, appelé *query set*, dans lequel les échantillons font l'objet d'une prédiction fondée sur l'observation des exemples du support set. Un épisode $\mathcal{E} \triangleq \{\mathcal{S}, \mathcal{Q}\}$ se compose d'un *support set* et du *query set* associé. L'entraînement dans ce contexte consiste à mettre à jour les poids du modèle en fonction de la prédiction sur les éléments du *query set*.

2.2 Modèle

Nous utilisons un modèle de réseaux prototypiques (Snell *et al.*, 2017, *prototypical networks*) avec un apprentissage épisodique afin de combiner méta-apprentissage et FSL. Notre modèle se compose de trois modules, comme illustré par la figure 1.

Module d'encodage. Étant donné une phrase $x = \{w_1, \ldots, w_L\}$, de longueur L, l'objectif de ce module est de construire une représentation e_i de chaque mot $w_i \in x$ dans un espace de dimension d. Nous utilisons le modèle de langue BERT (*Bidirectional Encoder Representations from Transformers* (Devlin *et al.*, 2019)) comme encodeur, qui est un modèle de type transformer composé de 12 couches entraînables. Soit $H_i = [h_i^1, h_i^2, \ldots, h_i^{12}]$, la sortie pour le mot w_i avec $h_i^j \in \mathbb{R}^d$.

Le principal objectif de notre travail est d'étudier plusieurs options d'exploitation de ces différentes couches afin d'obtenir des plongements de mots plus pertinents pour la résolution de la tâche de détection d'évènements. Plus précisément, partant de l'option par défaut appelée **BERT**, nous avons exploré cinq configurations de sélection et de combinaison des couches de l'encodeur :

— **BERT** : utilise la sortie de la dernière couche de BERT comme plongement du mot $e_i = h_i^{12}$. À la suite de Devlin *et al.* (2019), c'est l'option adoptée de façon générale pour les tâches de traitement automatique de langues utilisant BERT, en particulier par Cong *et al.* (2021) et Lai *et al.* (2021) dans notre contexte le plus proche.

— **Average** : moyenne des représentations sur m couches consécutives comme plongement du mot w_i. $e_i = \frac{1}{m}\sum_{k=1}^{m} h_i^k$ ou $e_i = \frac{1}{m}\sum_{k=12-m+1}^{12} h_i^k$ suivant que les couches sont associées à partir de la première ou de la dernière.

— **Max-pool** : max-pooling sur chaque dimension d sur m couches consécutives de sortie. Le p-ième élément du plongement e_i est donné par : $(e_i)_p = \max((h_i^1)_p, \ldots, (h_i^m)_p)$

— **Concat** : concaténation des sorties de m couches consécutives de BERT
$e_i = [h_i^1 \,||\, h_i^2 || \ldots || h_i^m]$ ou $e_i = [h_i^{12} \,||\, h_i^{11} \,||\, \ldots \,||\, h_i^{12-m+1}]$

— **Weighted** : combinaison linéaire des 12 couches de BERT. $e_i = \sum_{k=12}^{12} \alpha^k h_i^k$, où les α^k sont initialisés aléatoirement et appris.

— **ATT** : combinaison linéaire sur chaque dimension par mécanisme d'attention, l'objectif étant d'identifier pour chaque dimension la ou les couches les plus importantes. Le p-ième élément de e_i est donné par $(e_i)_p = \sum_{k=1}^{12} \alpha^k (h_i^k)_p$ où les α^k sont obtenus par apprentissage à partir d'une combinaison linéaire sur les 12 couches et d'une normalisation par un softmax.

Module prototypique. Son objectif est de construire un représentant de chaque classe appelé prototype, le but étant ensuite de classifier de nouveaux exemples en fonction de leur similarité à ces prototypes. Nous prenons comme prototype pour chaque classe la moyenne des exemples du support set appartenant à cette classe, comme proposé par Snell *et al.* (2017). Puisque que nous utilisons le format BIO en entrée, nous construisons un prototype pour les classes B et I ainsi que pour la classe O (dite classe *nulle*), qui désigne les mots qui ne sont déclencheurs d'aucun évènement. Au total, nous avons donc $2N + 1$ prototypes pour un épisode composé de N classes.

Module de classification. Ce module permet de classifier les mots d'une instance du query set en fonction de leur similarité aux prototypes. Pour une séquence donnée, on calcule la probabilité d'appartenance à chaque classe en fonction de leur similarité par rapport aux prototypes. Le modèle est ensuite entraîné en utilisant l'entropie croisée sur cette distribution de probabilités comme fonction objectif.

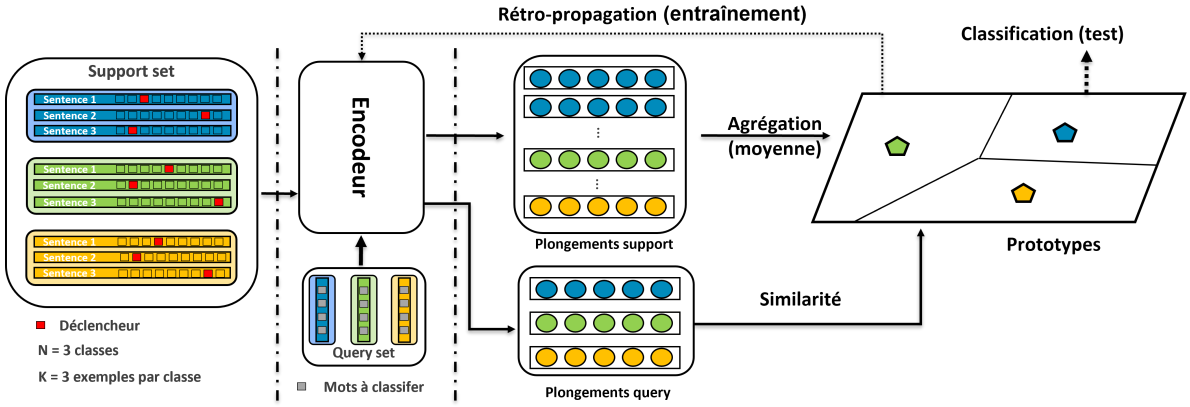


FIGURE 1 – Vue d'ensemble du modèle

3 Expérimentations et résultats

3.1 Cadre expérimental

Données d'évaluation. Nos expériences ont été réalisées sur le corpus FewEvent mis en place par Deng *et al.* (2020) pour la détection d'évènements à partir de peu d'exemples. Ce corpus est composé de 70 852 mentions d'évènements, en langue anglaise, réparties en 100 types. Nous utilisons le même découpage que Cong *et al.* (2021) à des fins de comparaison. Ce découpage comprend 80 types dans l'ensemble d'apprentissage, 10 types dans l'ensemble de test et les 10 autres dans l'ensemble de validation. La figure 2 montre deux mentions d'évènement des types *Election* et *Collaboration* où les mots en couleurs désignent les déclencheurs événementiels présents dans les phrases.

Paramètres du modèle. Nous utilisons le modèle pré-entraîné BERT-base comme encodeur de départ. Afin d'évaluer l'impact des modifications sur l'encodeur, nous utilisons deux modèles présentés dans Cong *et al.* (2021) : **Proto dot**, un modèle prototypique utilisant le produit scalaire comme fonction de similarité, qui est notre modèle de référence, et **PA-CRF**, une amélioration du modèle précédent utilisant des CRF (*Conditional Random Fields*) (Lafferty *et al.*, 2001) pour estimer les probabilités de transition entre les différents étiquettes BIO comme proposé par Hou *et al.* (2020). Ce modèle PA-CRF est la contribution principale de Cong *et al.* (2021) et constitue le meilleur modèle

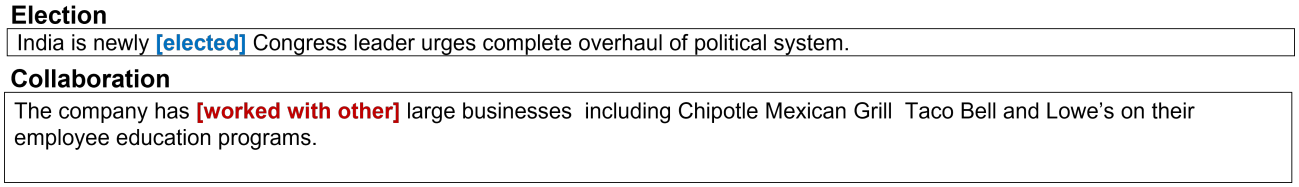


FIGURE 2 – Exemples de mentions d'évènements : les mots entre [] sont les déclencheurs évènementiels associés aux types en gras

de l'état de l'art actuel. Nous prenons en entrée des phrases de 128 mots (avec padding si besoin) avec des plongements de dimension 768 et nous entraînons le modèle avec un taux d'apprentissage de 10^{-5}.

Méthode d'évaluation. Pour l'évaluation, nous construisons 3 000 épisodes $\mathcal{E}^i \triangleq \{\mathcal{S}^i, \mathcal{Q}^i\}$ avec N types d'évènements tirés aléatoirement pour chaque épisode. Nous sélectionnons ensuite k exemples par classe dans le support set et un exemple par classe dans le query set. Les exemples du support set servent à construire les prototypes et les exemples du query set sont classifiés en fonction de leur similarité à ces prototypes. Nous considérons qu'un déclencheur d'évènement est correct si son type et sa position dans la phrase sont correctement prédits, comme dans les travaux précédents en détection d'évènements (Cong *et al.*, 2021; Cui *et al.*, 2020; Liu *et al.*, 2018a).

3.2 Résultats et discussion

Nous adoptons la micro F1-mesure pour évaluer les performances et nous rapportons les moyennes et les écarts types sur 5 essais dans le tableau 1 avec différentes valeurs de N et k. Pour les encodeurs **Average**, **Concat** et **Max-pool**, nous ne prenons en considération que les 4 dernières couches pour les résultats rapportés dans le tableau. Nous comparons notre modèle au modèle de Cong *et al.* (2021), qui donne la meilleure performance actuelle sur la même tâche. Nous avons ré-implémenté la version de Cong *et al.* (2021) correspondant à la ligne **BERT** et nous avons également rapporté les résultats fournis dans leur article (**BERT [Cong]**).

Le tableau 1 montre d'abord que, quel que soit le modèle utilisé (Proto-dot ou PA-CRF), toutes les modifications de l'encodeur, hormis la configuration **Concat** pour la condition 10 ways de PA-CRF, permettent d'améliorer de façon significative les performances par rapport à l'encodeur BERT classique. Une meilleure exploitation des informations du modèle BERT permet donc de dépasser les améliorations apportées par le modèle plus complexe de Cong *et al.* (2021), représentant l'état de l'art actuel.

Parmi les différentes stratégies testées, celles permettant au système d'apprendre automatiquement les poids pour combiner les différentes couches donnent généralement de meilleurs résultats, la stratégie **Weighted** s'avérant la meilleure dans presque tous les cas.

Enfin, le fait de retrouver les gains en F1-mesure observés pour le modèle Proto-dot au niveau du modèle PA-CRF, une version plus élaborée du premier, montre que les améliorations proposées sont complémentaires par rapport à celles pouvant être apportées aux autres modules (modules prototypique et de classification).

Impact de la formulation épisodique *N ways k shots*. De façon attendue, on remarque que la difficulté de la tâche augmente lorsque le nombre de classes (N) augmente et, de façon complémentaire,

Modèle	Encodeur	5 ways 5 shots	5 ways 10 shots	10 ways 5 shots	10 ways 10 shots
Proto-dot	BERT [Cong]	58,82 ± 0,88	61,01 ± 0,23	55,01 ± 1,62	58,78 ± 0,88
	BERT	61,22 ± 0,90	60,84 ± 1,58	58,14 ± 1,69	59,85 ± 2,01
	Average	64,34 ± 1,94	65,37 ± 0,66	61,85 ± 2,05	63,93 ± 1,08
	Max-pool	64,10 ± 1,78	65,80 ± 0,91	61,15 ± 1,51	63,37 ± 1,03
	Concat	61,99 ± 0,46	61,94 ± 0,97	57,47 ± 1,65	59,02 ± 1,39
	Weighted	65,62 ± 1,55	**67,15** ± 0,88*	**62,63** ± 1,18*	**65,22** ± 0,98*
	ATT	**65,64** ± 0,90	65,63 ± 0,46	62,22 ± 0,52	64,23 ± 0,99
PA-CRF	BERT [Cong]	62,25 ± 1,42	64,45 ± 0,49	58,48 ± 0,68	61,54 ± 0,89
	BERT	63,63 ± 2,01	63,66 ± 1,54	62,11 ± 1,58	62,47 ± 1,29
	Average	65,09 ± 0,40	66,70 ± 0,45	62,32 ± 1,51	65,38 ± 1,71
	Max-pool	63,95 ± 1,99	66,94 ± 1,20	61,74 ± 1,95	64,77 ± 1,84
	Concat	64,30 ± 1,99	64,31 ± 1,80	62,01 ± 1,28	61,88 ± 1,05
	Weighted	**66,26** ± 1,16*	**66,97** ± 0,95*	**63,90** ± 1,23*	**67,21** ± 1,27*
	ATT	63,65 ± 1,35	66,40 ± 1,03	62,41 ± 1,73	64,32 ± 1,64

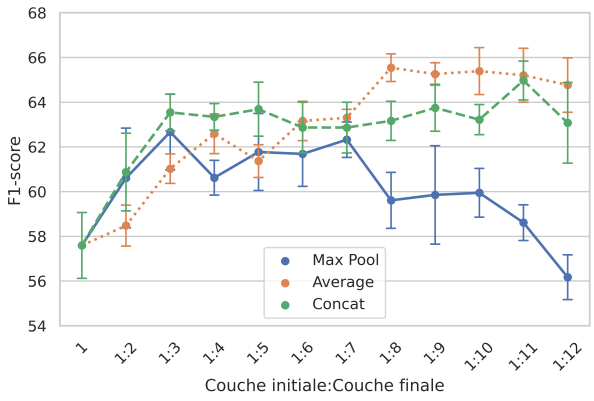
TABLE 1 – Résultats : moyenne et écart-type de la micro F1-mesure sur 5 essais. La meilleure performance en moyenne est indiquée **en gras**, la deuxième est soulignée. * indique que la différence entre le meilleur modèle et le deuxième est statiquement significative, en utilisant le test de significativité de Dror *et al.* (2019).

que les résultats sont meilleurs avec un plus grand nombre d'exemples annotés (k). En revanche, le nombre d'épisodes d'évaluation n'a pas d'influence significative sur les résultats à partir d'un certain niveau. Nous avons ainsi vérifié, en faisant varier le nombre d'épisodes de test (avec l'encodeur **Average**) entre 500 et 5 000, que les écarts de scores observés sont de moins de 0,5 point, bien en dessous des écart-types observés dans le tableau des résultats.

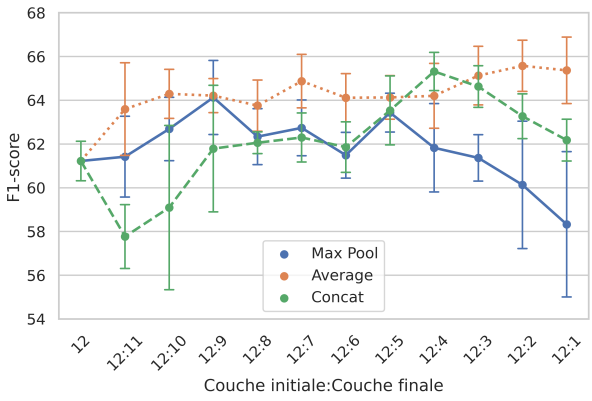
Impact de la formulation BIO. Nous avons utilisé le format BIO pour la détection et la classification des déclencheurs événementiels mais ce format pose plusieurs problèmes, en particulier dans un contexte de l'apprentissage à partir de peu d'exemples. Tout d'abord, les méthodes d'apprentissage à partir de peu d'exemples ont des difficultés pour estimer efficacement les transitions entre étiquettes comme il est d'usage pour les problèmes d'annotation de séquences. C'est d'ailleurs pour cette raison que Hou *et al.* (2020) proposent une méthode spécifique pour apprendre ces transitions dans un contexte d'apprentissage avec peu d'exemples.

Ce format implique par ailleurs de construire un prototype pour la classe "O", aussi appelée classe *nulle* dans le cadre d'un réseau prototypique. Or, cette classe possède par définition une cohérence très faible car elle est constituée de mots sans lien particulier sur le plan sémantique. La représentativité de son prototype est donc aussi très faible, ce qui tend à perturber les décisions de classification évènementielle des mots. Enfin, les classes "I" posent aussi problème dans la mesure où elles correspondent souvent à des prépositions dans des verbes à particules (*phrasal verbs*). Le modèle a donc tendance à les confondre avec des étiquettes I d'autres classes ou avec des mots de la classe nulle. Par ailleurs, ces étiquettes I n'interviennent que dans le cas de déclencheurs multi-mots, qui sont peu représentés dans les corpus et conduisent donc à des prototypes peu fiables.

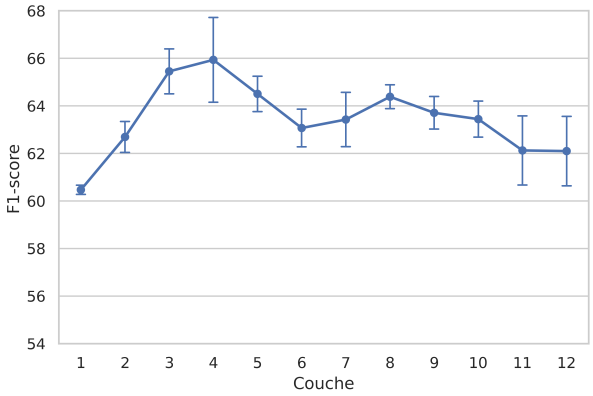
Analyse couche par couche. Pour compléter l'analyse, nous avons réalisé des tests afin de déterminer l'influence du nombre de couches sélectionnées pour les encodeurs **Average**, **Concat** et **Max-pool**. Nous rapportons à la figure 3 les résultats obtenus par ces trois modèles pour la tâche *5 ways 5 shots*



(a) Premières couches



(b) Dernières couches



(c) Performance couche par couche

FIGURE 3 – Les figures 3a et 3b présentent l'influence du nombre de couches sélectionnées pour les encodeurs **Average**, **Concat**, et **Max-pool** sur les performances du modèle. La figure 3c présente les performances du modèle pour une couche isolée.

en prenant en compte n couches successives en partant de la première couche (courbes à gauche) ou de la dernière (courbes à droite).

Nous constatons que l'encodeur **Average** est plus stable que les deux autres et que ses performances ont tendance à augmenter régulièrement avec le nombre de couches. Ses meilleurs résultats sont par ailleurs compétitifs par rapport à la stratégie **Weighted**, ce qui montre que la prise en compte de toutes les couches reste intéressante, même à l'aide d'une combinaison simple comme la moyenne. Les autres stratégies ne permettent pas, quant à elles, d'exploiter toutes les informations. C'est particulièrement notable pour **Max-pool** qui tend probablement à lisser de plus en plus les éléments saillants quand le nombre de couches augmente. **Concat** crée pour sa part des représentations intermédiaires de taille importante qui sont sans doute trop peu sélectives du point de vue de leur exploitation par le modèle.

Par ailleurs, les figures 3a et 3b montrent que les dernières couches de BERT semblent plus intéressantes que les premières dans notre contexte. Ce constat peut traduire la présence intrinsèque d'informations plus utiles pour la tâche visée dans ces couches ou simplement s'expliquer par une influence plus importante de l'apprentissage liée à la tâche à leur niveau du fait de leur plus grande proximité vis-à-vis de la sortie du modèle.

Enfin, la figure 3c rapporte le cas limite de la prise en compte d'une seule couche, avec, de la première à la dernière couche, une forte augmentation des résultats jusqu'à la couche 4, qui atteint

une performance comparable à **Average** mais avec une variance plus importante, puis une baisse progressive.

4 Conclusions et perspectives

Dans cet article, nous avons étudié différentes façons de mieux exploiter les informations contenues dans le modèle pré-entraîné BERT pour la tâche de détection d'évènements à partir de peu d'exemples. Nous avons montré que les améliorations apportées par nos propositions permettent de dépasser les résultats de l'état de l'art sur cette tâche.

Par la suite, nous envisageons de poursuivre les améliorations de l'encodeur en étudiant d'autres modèles de représentation et en enrichissant ces représentations par des connaissances extérieures ou des listes d'exemples de déclencheurs événementiels. Par ailleurs, nous étudierons la combinaison de ces améliorations de l'encodeur avec des améliorations complémentaires du module prototypique et du module de classification.

Références

AHN D. (2006). The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, p. 1–8, Sydney, Australia : Association for Computational Linguistics.

BRONSTEIN O., DAGAN I., LI Q., JI H. & FRANK A. (2015). Seed-Based Event Trigger Labeling : How far can event descriptions get us ? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, p. 372–376, Beijing, China : Association for Computational Linguistics. DOI : 10.3115/v1/P15-2061.

CHEN J., LIN H., HAN X. & SUN L. (2021). Honey or Poison ? Solving the Trigger Curse in Few-shot Event Detection via Causal Intervention. *arXiv :2109.05747 [cs]*. arXiv : 2109.05747.

CONG X., CUI S., YU B., LIU T., YUBIN W. & WANG B. (2021). Few-Shot Event Detection with Prototypical Amortized Conditional Random Field. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, p. 28–40, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.findings-acl.3.

CUI S., YU B., LIU T., ZHANG Z., WANG X. & SHI J. (2020). Edge-Enhanced Graph Convolution Networks for Event Detection with Syntactic Relation. In *Findings of the Association for Computational Linguistics : EMNLP 2020*, p. 2329–2339, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.findings-emnlp.211.

DENG S., ZHANG N., KANG J., ZHANG Y., ZHANG W. & CHEN H. (2020). Meta-Learning with Dynamic-Memory-Based Prototypical Network for Few-Shot Event Detection. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, p. 151–159, Houston TX USA : ACM. DOI : 10.1145/3336191.3371796.

DENG S., ZHANG N., LI L., HUI C., HUAIXIAO T., CHEN M., HUANG F. & CHEN H. (2021). OntoED : Low-resource Event Detection with Ontology Embedding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, p. 2828–2839, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-long.220.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2019). BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv :1810.04805 [cs]*. arXiv : 1810.04805.

DROR R., SHLOMOV S. & REICHART R. (2019). Deep dominance - how to properly compare deep neural models. In A. KORHONEN, D. R. TRAUM & L. MÀRQUEZ, Éds., *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1 : Long Papers*, p. 2773–2785 : Association for Computational Linguistics. DOI : 10.18653/v1/p19-1266.

FINN C., ABBEEL P. & LEVINE S. (2017). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv :1703.03400 [cs]*. arXiv : 1703.03400.

HOU Y., CHE W., LAI Y., ZHOU Z., LIU Y., LIU H. & LIU T. (2020). Few-shot Slot Tagging with Collapsed Dependency Transfer and Label-enhanced Task-adaptive Projection Network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 1381–1393, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.acl-main.128.

LAFFERTY J. D., MCCALLUM A. & PEREIRA F. C. N. (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.

LAI V., DERNONCOURT F. & NGUYEN T. H. (2021). Learning Prototype Representations Across Few-Shot Tasks for Event Detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, p. 5270–5277, Online and Punta Cana, Dominican Republic : Association for Computational Linguistics.

LAI V. D. & NGUYEN T. (2019). Extending Event Detection to New Types with Learning from Keywords. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, p. 243–248, Hong Kong, China : Association for Computational Linguistics. DOI : 10.18653/v1/D19-5532.

LAI V. D., NGUYEN T. H. & DERNONCOURT F. (2020). Extensively Matching for Few-shot Learning Event Detection. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, p. 38–45, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.nuse-1.5.

LAKE B. M., SALAKHUTDINOV R. & TENENBAUM J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, **350**(6266), 1332–1338. DOI : 10.1126/science.aab3050.

LI Q., JI H. & HUANG L. (2013). Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 73–82, Sofia, Bulgaria : Association for Computational Linguistics.

LIU S., CHENG R., YU X. & CHENG X. (2018a). Exploiting Contextual Information via Dynamic Memory Network for Event Detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, p. 1030–1035, Brussels, Belgium : Association for Computational Linguistics. DOI : 10.18653/v1/D18-1127.

LIU X., LUO Z. & HUANG H. (2018b). Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, p. 1247–1256, Brussels, Belgium : Association for Computational Linguistics. DOI : 10.18653/v1/D18-1156.

NGUYEN T. H., CHO K. & GRISHMAN R. (2016). Joint Event Extraction via Recurrent Neural Networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, p. 300–309, San Diego, California : Association for Computational Linguistics. DOI : 10.18653/v1/N16-1034.

NGUYEN T. H. & GRISHMAN R. (2015). Event Detection and Domain Adaptation with Convolutional Neural Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, p. 365–371, Beijing, China : Association for Computational Linguistics. DOI : 10.3115/v1/P15-2060.

NGUYEN T. H. & GRISHMAN R. (2018). Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18 : AAAI Press.

NICHOL A., ACHIAM J. & SCHULMAN J. (2018). On first-order meta-learning algorithms.

RAMSHAW L. & MARCUS M. (1995). Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

RAVI S. & LAROCHELLE H. (2017). Optimization as a model for few-shot learning. In *ICLR*.

SANTORO A., BARTUNOV S., BOTVINICK M., WIERSTRA D. & LILLICRAP T. (2016). Meta-learning with memory-augmented neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, p. 1842–1850 : JMLR.org.

SHEN S., WU T., QI G., LI Y.-F., HAFFARI G. & BI S. (2021). Adaptive Knowledge-Enhanced Bayesian Meta-Learning for Few-shot Event Detection. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, p. 2417–2429, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.findings-acl.214.

SNELL J., SWERSKY K. & ZEMEL R. (2017). Prototypical networks for few-shot learning. In I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems*, volume 30 : Curran Associates, Inc.

SUNG F., YANG Y., ZHANG L., XIANG T., TORR P. H. S. & HOSPEDALES T. M. (2018). Learning to compare : Relation network for few-shot learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, p. 1199–1208.

VINYALS O., BLUNDELL C., LILLICRAP T., KAVUKCUOGLU K. & WIERSTRA D. (2016). Matching networks for one shot learning. In D. LEE, M. SUGIYAMA, U. LUXBURG, I. GUYON & R. GARNETT, Éds., *Advances in Neural Information Processing Systems*, volume 29 : Curran Associates, Inc.

WALKER C., STRASSEL S. & JULIE MEDERO K. M. (2006). *ACE 2005 Multilingual Training Corpus*. LDC corpora. Linguistic Data Consortium. DOI : 10.35111/mwxc-vh88.

YAN H., JIN X., MENG X., GUO J. & CHENG X. (2019). Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, p. 5766–5770, Hong Kong, China : Association for Computational Linguistics. DOI : 10.18653/v1/D19-1582.

YAN W., YAP J. & MORI G. (2015). Multi-task transfer methods to improve one-shot learning for multimedia event detection. In X. XIE, M. W. JONES & G. K. L. TAM, Éds., *Proceedings of the British Machine Vision Conference (BMVC)*, p. 37.1–37.13 : BMVA Press. DOI : 10.5244/C.29.37.

ZHANG H., WANG H. & ROTH D. (2021). Zero-shot Label-Aware Event Trigger and Argument Classification. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, p. 1331–1340, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.findings-acl.114.