Actes de la journée d'étude sur le TAL frugal et la RI frugale

Avec le soutien de l'ATALA et du GDR TAL

Éditeurs

Caio Corro
Sorbonne Université, ISIR

Gaël Lejeune
Sorbonne Université, STIH

25 Janvier 2024
Maison de la recherche, 28 rue Serpente, 75006 Paris

Préface

Cette journée d'étude vise à réunir les collègues s'intéressant à la frugalité dans les systèmes de TAL, le thème de la frugalité pouvant se décliner selon les trois axes suivant :

— Frugalité des données d'entraînement (taille des corpus, langues et domaines peu dotés, few-shot learning, apprentissage avec des données synthétiques, etc),
— Frugalité des coûts d'entraînement (matériel, temps de calculs, coût en ingénierie, etc)
— Frugalité des modèles (taille des modèles, utilisation en local et/ou environnement restreint, etc).

Remerciements

Nous remercions l'ATALA et le GDR TAL pour le soutien financier. Nous remercions les auteurs et autrices des résumés qui rendent cette journée d'étude possible.

Programme de la journée

	Accueil & café (Hall du 3ème étage)
10:30 - 11:00	**Entre performance et frugalité en TAL : Approches pour la réduction de la taille des (L)LMs** Xavier Pillet, Anastasia Volkova, Nicolas Greffard, Richard Dufour
11:00 - 12:00	Présentation invitée **Introduction a la RI neuronale et aux modeles SPLADE** Stéphane Clinchant
	Buffet (Hall du 3ème étage)
13:30 - 14:00	**Essai d'étude quantitative des schémas de composition nominale en grec ancien : la quête de données de qualité** Louis Jourdain
14:00 - 14:30	**Décodage paresseux : décodage contraint pour l'extraction d'informations** Arthur Hemmer, Mickaël Coustaty, Nicola Bartolo, Jérôme Brachat, Jean-Marc Ogier
14:30 - 15:00	**Où la frugalité rejoint l'éthique : utilisation de données synthétiques pour la reconnaissance d'entités cliniques** Nicolas Hiebel, Olivier Ferret, Karën Fort, Aurélie Névéol
	Pause café (Hall du 3ème étage)
15:30 - 16:00	**Practical Considerations for implementing Sparse Tensor Manipulation over arbitrary Semirings** Lucas Ondel Yang
16:30 - 17:00	**Détection d'événements à partir de peu d'exemples par méta-apprentissage** Aboubacar Tuo, Olivier Ferret, Romaric Besançon, Julien Tourille
17:00 - 17:30	**De l'influence des représentations des mots pour la détection d'anomalies textuelles dans un cadre *k-classes-out*** Alicia Breidenstein, Matthieu Labeau

Table des matières

ENTRE PERFORMANCE ET FRUGALITÉ EN TAL : APPROCHES POUR LA RÉDUCTION DE LA TAILLE DES (L)LMS

Xavier Pillet
Nantes Université, LS2N & Valeuriad
pillet.xavier@valeuriad.fr

Anastasia Volkova
Inria Lyon, CITI
anastasia.volkova@inria.fr

Nicolas Greffard
Valeuriad
greffard.nicolas@valeuriad.fr

Richard Dufour
Nantes Université, LS2N
richard.dufour@ls2n.fr

Introduction

Ces dernières années, les performances des tâches de traitement automatique du langage (TAL) ont largement progressé notamment au travers des modèles de langues pré-entraînés. Ces avancées se sont traduites par une augmentation importante de la taille de ces modèles [1], les rendant de plus en plus coûteux en matériel et en temps de traitement, que ce soit au niveau de leur entraînement et/ou de leur utilisation en inférence. Pour pallier ces problèmes, de nombreux travaux ont émergé autour de l'optimisation des modèles de langue pré-entraînés. Cet article résume les principales approches actuelles sur la réduction du coût mémoire des paramètres (*e.g.* les poids et les biais), considérant le fait que la RAM des GPU est un goulot d'étranglement majeur pour les modèles les plus larges [2].

Méthodes

Plusieurs approches existent, notamment celles qui consistent à diminuer le nombre de paramètres (élagage) et celles consistant à en réduire la précision des formats numériques (quantification).

Élagage

L'élagage (*pruning* en anglais) consiste généralement à forcer certains paramètres à 0. Ces nombreux 0 ne sont vraiment intéressants que s'ils exploitent avantageusement les possibilités logicielles (*e.g.* la structure de données du tenseur parcimonieux) et les accélérations matérielles, ce qui est plus facile si l'emplacement des 0 est connue à l'avance. Nous distinguons alors l'élagage non-structuré (l'emplacement des 0 est *a priori* aléatoire) du structuré.

Une des premières techniques non-structurées consiste à utiliser l'information du second ordre. Après entraînement, nous pouvons supprimer les petits paramètres qui se situent dans les directions propres de la *hessienne* à petites valeurs propres (*i.e.* ses directions propres plates) [3]. Comme le gradient ne change pas beaucoup le long de ces directions et qu'il est presque nul à la fin de l'entraînement, cela signifie que la fonction de coût y est aussi stable et que les paramètres peuvent être légèrement modifiés le long de ces directions, sans impacter la performance du modèle.

Une autre méthode d'élagage est celle du *ticket gagnant*. Un ticket gagnant est une combinaison de paramètres qui contient un grand pourcentage de 0, à la fin de l'entraînement. Cela peut être obtenu au prix de nombreux réentraînements coûteux [4]. Cette parcimonie est non-structurée, mais elle peut atteindre jusqu'à 90 % de 0 pour les modèles TAL [4]. Ceci suggère une sur-paramétrisation initiale importante et que l'on devrait pouvoir, dès l'entraînement, utiliser moins de paramètres. Cela est en effet possible avec l'heuristique d'entraînement des *tickets truqués*, où de nombreux paramètres sont initialisés et maintenus à 0 au cours de l'entraînement, sauf ceux à la dérivée (*i.e.* l'erreur) la plus forte (donc au meilleur potentiel pour l'apprentissage) qui sont insérés [5].

On peut aussi concevoir des couches nativement parcimonieuses, ce qui permet d'obtenir en plus une parcimonie structurée. Par exemple, utiliser des matrices de poids creuses dans les couches linéaires. Si l'on souhaite rendre un transformer complètement parcimonieux, nous devons concevoir une version parcimonieuse à la fois des têtes d'attention et du réseau direct en sortie. Pour les têtes d'attention, il y a le *longformer* [6], qui utilise aussi des matrices creuses. Pour le réseau direct, une version parcimonieuse est la *mixture d'experts* [7] fondée sur une softmax parcimonieuse. Combinées, ces deux couches forment un transformer complètement parcimonieux, pouvant bénéficier d'accélération matérielle et traiter de longs documents, comme dans le cas du *terraformer* [8].

Quantification

La quantification (*quantization* en anglais) est une approche de réduction de la taille des modèles qui consiste à réduire la précision numérique des paramètres pour optimiser la consommation mémoire et le temps de calcul. La paradigme existant pour l'entraînement est de stocker les paramètres en format à virgule flottante (*floating point* (FP) en anglais) de taille 32 bits (FP32), mais de faire les multiplications sur GPU en FP16 [9]. Cette précision mixte FP32/FP16 a permis l'émergence des gigamodèles de langue (*Large Language Models* en anglais, ou LLMs) [1]. Pour l'inférence on a pas besoin de la virgule flottante et on peut tirer partie de la vitesse de calcul plus rapide des entiers, notamment à 8 bits (INT8) [10]. En revanche, le passage des flottants vers les entiers après entraînement (*post-training quantization* en anglais) n'est pas trivial et influe sur les performances. Cela peut-être fait en utilisant l'information de la hessienne, de façon similaire à l'élagage [10]. Cependant, si l'on dispose des données d'entraînement, on peut faire du ré-entraînement en quantifiant (*quantization aware training* en anglais), ce qui est souvent plus performant [11].

L'apparition de nouvelles architectures GPU avec un support matériel d'opérations au format FP8 [12] rend possible l'entraînement directement en précision mixte FP16/FP8. La quantification en INT8 n'apparaît donc plus nécessaire [2]. Cependant, les entiers sont meilleurs pour représenter des nombres uniformément distribués et sont plus rapides en calcul, alors que les flottants représentent mieux les nombres entre 0 et 1 par exemple [13]. Or, un effet de seuil semble apparaître pour les modèles suffisamment larges (de l'ordre la dizaine de milliards de paramètres) qui voient apparaître une forte asymétrie dans la distribution des valeurs des poids des couches d'attention. Cette asymétrie est corrélée à une amélioration importante des performances et rend les poids mieux représentables par des FP8 [2]. Donc pour les modèles en dessous du milliard de paramètres, quantifier en INT8 reste pertinent, si l'on lisse la distribution des poids des attentions, grâce à la régularisation L_2 par exemple [13]. Au delà, mieux veut utiliser les FP8 pour les LLMs, pour conserver le phénomène d'émergence [13]. Des travaux en cours étudient un nouveau format numérique, à base de micro-exposants (MX), qui factorise en commun les exposants des tenseurs et sous-tenseurs, ce qui permettrait de bien représenter ces distributions asymétriques, tout en étant efficace en termes de stockage [14].

Matrices à bas rang

S'inspirant de l'algèbre linéaire pour le calcul haute performance, afin d'économiser de la RAM, on peut approcher les grandes matrices avec de beaucoup plus petites, grâce à des approximations à bas rang. Cette nouvelle représentation est efficace en termes de consommation de mémoire et on peut aussi entraîner directement les décompositions de tenseurs. Cela est notamment fait dans les smartphones avec les MobileNets [15].

Adaptateurs

Pour faire face au coût important du ré-entraînement des LLMs, la solution des *adaptateurs* a été proposée : elle consiste à intercaler des couches adaptatrices en leur sein [16]. On peut alors ré-entraîner uniquement les matrices de paramètres de ces couches adaptatrices, sans toucher au reste du modèle, ce qui permet de pouvoir n'utiliser qu'un LLM générique et de l'adapter à plusieurs domaines différents juste en changeant les adaptateurs. Comme ces adaptateurs rajoutent des sous-couches, ils ralentissent un peu les calculs et rajoutent des paramètres. L'utilisation de matrices adaptatrices à bas rang permet de minimiser cet impact, ce qui donne les méthodes LoRA (de l'anglais *Low Rank Adapter*) [17], ou QLoRA pour les modèles quantifiés [18].

Distillation

La *distillation* consiste à entraîner un petit réseau à imiter le comportement d'un plus large [19]. Elle est coûteuse en termes de ré-entraînement mais parvient à comprimer efficacement certains modèles [20] et permet de tirer indirectement partie de leurs corpus d'entraînement parfois privés et onéreux. Certains fournisseurs s'en protègent en l'interdisant *via* des licences, telle que celle de LLaMA 2. La distillation peut être mixée avec de la quantification [21].

Défis méthodologiques

L'étude des différentes méthodes de réduction de (L)LM pose certains défis méthodologiques. Premièrement, la plupart utilisent un ré-entraînement ou affinage final. Hors, beaucoup sont sous-entraînés [22]. Cela pose alors la question de l'origine de la restauration de la performance après réduction : est-t-elle due à l'efficacité de la méthode, ou au fait que l'entraînement était poursuivi ? Deuxièmement, le coût parfois prohibitif pour la recherche académique de leur entraînement est un autre obstacle à la reproductibilité scientifique des résultats, ce qui limite leur pertinence. Finalement, de plus en plus de métriques sur des tâches différentes sont utilisées, dont la pertinence et la qualité est discutable. Par exemple l'une des performances émergeantes est sur une tâche d'addition à 8 bits, ce qui est ridicule compte tenu de la taille du modèle : pour 8 bits, il y a $2^8 \times 2^8 = 65536$ cas d'additions possibles. Donc avec plus de 7 milliards de paramètres, le modèle peut avoir tout appris par cœur, surappris ces additions [2].

Conclusion et perspectives

Diverses approches de réduction de la taille des grands modèles de langues ont été étudiées. De nombreuses questions restent en suspens, telles que : Si l'on arrive à élaguer à ce point les modèles, a-t-on réellement besoin d'autant de paramètres ? Quelle est la performance maximale que l'on peut atteindre avec les modèles frugaux ? Comment la quantification, par exemple, influence-t-elle spécifiquement certaines tâches ou compétences langagières des modèles ?

Références

[1] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama : Open and efficient foundation language models. *arXiv preprint arXiv :2302.13971*, 2023.

[2] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 () : 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv :2208.07339*, 2022.

[3] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[4] Sai Prasanna, Anna Rogers, and Anna Rumshisky. When bert plays the lottery, all tickets are winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3208–3229, 2020.

[5] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery : Making all tickets winners. In *International Conference on Machine Learning*, pages 2943–2952. PMLR, 2020.

[6] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer : The long-document transformer. *arXiv preprint arXiv :2004.05150*, 2020.

[7] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks : The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv :1701.06538*, 2017.

[8] Sebastian Jaszczur, Aakanksha Chowdhery, Afroz Mohiuddin, Lukasz Kaiser, Wojciech Gajewski, Henryk Michalewski, and Jonni Kanerva. Sparse is enough in scaling transformers. *Advances in Neural Information Processing Systems*, 34 :9895–9907, 2021.

[9] Naveen Mellempudi, Sudarshan Srinivasan, Dipankar Das, and Bharat Kaul. Mixed precision training with 8-bit floating point. *arXiv preprint arXiv :1905.12334*, 2019.

[10] Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Q-bert : Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821, 2020.

[11] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. Q8bert : Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE, 2019.

[12] Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, et al. Fp8 formats for deep learning. *arXiv preprint arXiv :2209.05433*, 2022.

[13] Mart van Baalen, Andrey Kuzmin, Suparna S Nair, Yuwei Ren, Eric Mahurin, Chirag Patel, Sundar Subramanian, Sanghyuk Lee, Markus Nagel, Joseph Soriaga, et al. Fp8 versus int8 for efficient deep learning inference. *arXiv preprint arXiv :2303.17951*, 2023.

[14] Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, et al. Microscaling data formats for deep learning. *arXiv preprint arXiv :2310.10537*, 2023.

[15] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets : Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv :1704.04861*, 2017.

[16] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora : Low-rank adaptation of large language models. *arXiv preprint arXiv :2106.09685*, 2021.

[18] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora : Efficient finetuning of quantized llms. *arXiv preprint arXiv :2305.14314*, 2023.

[19] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert : smaller, faster, cheaper and lighter. *arXiv preprint arXiv :1910.01108*, 2019.

[20] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep ? *Advances in neural information processing systems*, 27, 2014.

[21] Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization. *arXiv preprint arXiv :1802.05668*, 2018.

[22] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv :2203.15556*, 2022.

ESSAI D'ÉTUDE QUANTITATIVE DES SCHÉMAS DE COMPOSITION NOMINALE EN GREC ANCIEN : LA QUÊTE DE DONNÉES DE QUALITÉ

Louis Jourdain

Parmi les langues peu dotées, les langues anciennes posent des difficultés particulières: si pour les langues les plus connues, des corpus de qualité ont été numérisés assez tôt, l'inventaire des textes disponibles en langues anciennes est quasiment clos et est souvent très hétérogène en terme d'époque , donc de langue, et de genre. On ne peut donc pas espérer, comme pour d'autres langues, augmenter la qualité des modèles et des systèmes que l'on construit en augmentant la quantité des données, et c'est donc sur la qualité des données employées et des représentations choisies qu'il faut travailler. La présente étude, en tentant de répondre à une problématique d'ordre linguistique tentera de présenter les étapes nécessaires à la constitution et à l'exploitation de données de qualité pour le grec ancien en insistant sur différents choix d'implémentation et leur impact sur la qualité des données. Sera également soulignée la nécessité de mettre en place des procédures pour évaluer la perte d'information et le bruit introduits dans les données à chaque étape du traitement qu'on leur fait subir.

1 Les problèmes linguistiques considérés :

La composition nominale est un procédé de création lexicale lors duquel deux termes autonomes dans le lexique se combinent morphologiquement et sémantiquement pour former un nouveau terme. Si la composition dans les langues modernes se résume souvent à la concaténation de deux termes (lave-vaisselle, toothbrush) et ne représente qu'une part minime du lexique (autour de 5%), la composition était un mode de formation très productif et vivant dans les langues indo-européennes. Elle est toutefois plus difficile à étudier à cause du nombre bien plus important de combinaisons possibles et du fait que des règles morpho-phonétiques déterminent la façon dont les membres doivent être combinés, notamment à leur jointure. De plus, les mots grecs ont plusieurs bases (nominatif et génitif pour les noms, thèmes de présent, aoriste, parfait ou racine nue pour les verbes). La composition pose donc un problème de sélection de base *stem selection* Il est fréquent aussi qu'un composé grec appartienne à une catégorie morphologique différente de celles des noms dont il est dérivé, ce qui est plus surprenant. En effet, selon le principe d'économie, on attendrait, et c'est ce qui se passe dans la plupart des cas, qu'un composé appartienne à la même classe morphologique et se décline de la même manière que son second membre. Or de nombreux composés présentent des suffixes. Dès lors, la formation d'un composé à partir d'une base n'est plus une opération triviale, et cela interroge la prédictibilité des résultats de la composition nominale en grec. Ce constat appelle deux questions précises visant à permettre une description plus claire du phénomène :

* Y-a-t-il des règles claires sur la manière de construire des noms composés en grec ancien ? Si oui quels sont ces schémas de construction ? Quelles sont les variables explicatives qui les organisent ? Concernant plus précisément le phénomène de resuffixation des composés nominaux , qui est typologiquement rare, Nathalie Rousseau (2016) a observé qu'il était fréquent notamment dans une classe particulière de composés, les composés hypostatiques (construits à partir d'un syntagme prépositionnel). * Mais ce phénomène concerne-t-il également les autres types de composés de façon significative ? Quel est précisément l'inventaire des suffixes impliqués ? Et y-a-t-il des règles régissant le choix du suffixe. Si oui, quels facteurs régissent ce choix ?

Si ces particularités de la composition nominale en grec ancien ont été relevés de longue date dans la littérature linguistique, à notre connaissance aucun travail n'a su proposer une description complète et organisée de ce système (les ouvrages se contentant d'expliquer la formation des termes attestés, sans se risquer à proposer des règles générales de formation des termes), ni quantifier de façon précise ces phénomènes de changement de classe et de resuffixation. Une analyse quantitative partant des données selon une démarche "bottom-up" pourrait permettre de considérer la question sous un nouvel angle. Toutefois, pour se faire, il faudrait disposer d'une liste de tous les composés grecs ou d'un moyen fiable de les détecter, car il est impossible d'étudier manuellement un corpus de plusieurs millions de mots. On présentera nos travaux de collecte et de traitement de données pour proposer une ébauche de réponses à ces questions linguistiques, insistant sur la démarche plus que les résultats. Le but minimal serait de pouvoir estimer la probabilité qu'un nom d'une classe morphologique A donne un composé de classe morphologique B (problème de la construction du composé) et la probabilité qu'un composé de classe morphologique A provienne d'une tête de classe morphologique B (problème de l'inférence de l'étymon).

2 Les données collectées et leur traitement à risque :

Il n'existe aucune base de donnée complète référençant tous les composés en grec ancien. Néanmoins, l'université de Palerme a lancé en 2017 le projet "*The Homeric Greek Compounds Project*", (http://homeric-compounds.scienzeumanistiche.eu/) qui présente près de 1500 formes richement annotées (premier et second membre isolés, les suffixes indiqués). Ce nombre de formes n'est toutefois pas suffisant pour construire un classifieur capable de repérer un composé de façon fiable, comme cela a été fait pour le sanskrit, mais avec dix fois plus de données (la tâche de détection des composés est une tâche complexe car c'est une évaluation hors domaine). Toutefois cette base de données annotée manuellement souffre d'irrégularité dans les normes d'annotation, des signes diacritiques, voire de données lacunaires pour certaines entrées. Un travail de nettoyage des données a été nécessaire. C'est donc une base de donnée présentant des informations sûres mais proposant un nombre limité de formes.

Analyser les schémas de suffixation aurait été impossible, si une ressource bien connue des hellénistes ne fournissait pas discrètement un corpus bien plus complet à traiter. En effet, dans l'édition papier du Bailly (1895), mais donc aussi sa récente numérisation : https://chaerephon.e-monsite.com/medias/files/bailly.html. les mots composés sont séparés par un point en haut, (par exemple ἐν·άλιος). Ce détail de format a rendu possible l'extraction de tous les termes composés. On obtient donc un corpus de 29505 composés sur un total de 110612 mots. Pour pouvoir estimer les distributions qui nous intéressent, il faut déterminer quelle est la tête de chaque composé et déterminer la classe morphologique et/ou le suffixe de la tête et du composé. Pour ces questions encore, les notices du dictionnaire peuvent fournir des informations précieuses que l'on peut extraire en effectuant un parsing des données html. Le dictionnaire propose régulièrement une note d'étymologie (par exemple pour ἐν·άλιος : Étym. ἐν, ἅλς.) mais les notices ne sont pas formatées de façon consistante dans un dictionnaire du 19 ième siècle. En absence de données étymologiques, on a dû procéder à une recherche (par simple distance de Levensthein) pour identifier la tête. Cette opération est susceptible d'introduire du bruit dans les données. En résumé le Bailly numérisé permet d'obtenir une base de donnée importante en taille mais qui ne présente pas d'information sûre sur la tête de chaque composé.

3 Inférence de classe et expériences menées :

Lors d'une première expérience, on a représenté la classe morphologique d'un mot par son suffixe. Une liste d'une cinquantaine de suffixes fréquents dans la composition a été établie à partir des annotations de la base de donnée des composés homériques. Un algorithme a été conçu pour repérer les suffixes dans les mots. Or il arrive que l'algorithme identifie comme suffixe une partie du radical (φίλος est un nom thématique en -ος pas un suffixé en -λος). Des garde-fous ont été implémentés pour éviter cela, mais en absence d'une correction effectuée sur la base d'un savoir linguistique, il semble impossible de délimiter base et suffixe sans employer un *tokenizer* morphologiquement adapté, qui à notre connaissance n'existe pas pour le grec. Les résultats de cette expérience ont montré que cette représentation n'était pas suffisante pour identifier les classes morphologiques et que certaines classes distinctes se retrouvaient mélangées dans les résultats. On a donc décrit dans une seconde expérience les classes morphologiques comme des tuples contenant la désinence du nominatif et celle du génitif, ce qui a permis d'effectuer des calculs à deux niveau, au niveau des classes morphologiques, (première question posée), et des sous classes marquées par des suffixes (second volet de l'enquête) Un algorithme associe chaque terme à sa classe, en identifiant le génitif du mot (facile dans la base de donnée homérique, *parsing* fiable à 90% dans le dictionnaire) puis cette information est croisée à la forme du lemme pour déduire la désinence de nominatif. Du bruit a pu être introduit dans cette étape de *matching*.

Une fois à notre disposition des informations sur la classe de chaque composé et sa tête, on a pu calculer les distributions souhaitées et identifier les schémas de recatégorisation (cas où le composé et sa tête sont différents) les plus productifs. On a également étudié la fréquence des schémas de recatégorisation (voir graphe ci dessous) et calculé l'entropie, l'entropie conditionnelle et l'information mutuelle sur ces distributions pour tenter d'évaluer si la connaissance de la classe morphologique de la tête était suffisante pour prédire correctement celle du composé. On pourra par la suite ajouter d'autres informations (tel que le genre, du composé et sa catégorie grammaticale, ou des informations sémantiques comme la distinction entre animés et inanimés pour voir si cela augmente la prédictibilité de la classe du composé, en d'autres termes si la connaissance de la classe de la tête et des propriétés morphologiques et sémantiques du terme que l'on veut dériver permettent de prédire avec certitude la forme du composé).

On peut finalement mener ces expériences sur trois corpus présentant différents avantages : le corpus homérique (annoté par des spécialistes, sûr, mais limité par la taille et le genre) le corpus enrichi avec certitude par les données étymologiques du Bailly le corpus complet des 29505 composés, sachant que certaines têtes pourraient avoir été mal identifiées Pour répondre à une question de recherche telle que celles que l'on a posées et décrire un système, on fait face à un impératif d'exhaustivité. Or lorsque les démarches d'enrichissement des données sont susceptibles d'introduire du bruit ou n'aboutissent pas pour certains exemples, il est raisonnable de les laisser de coté plutôt que de risquer de fausser l'étude. On aura montré que la réussite d'une telle étude repose sur la qualité et la quantité des

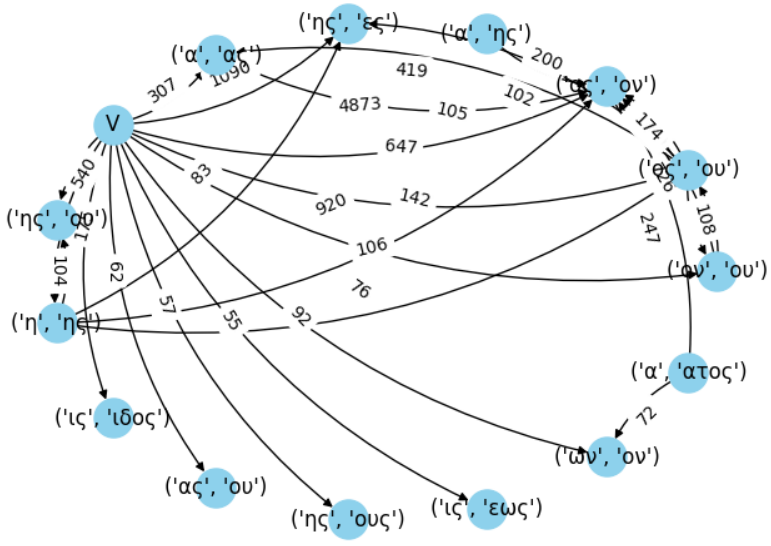


Figure 1: **Graphe représentant les changements de classes les plus fréquents (une flèche représentant le fait qu'un nom d'une certaine classe donne un composé d'une autre classe)**

données collectées, ce qui est un défi dans le cas des langues peu dotées. Le travail de l'informaticien sur les données est nécessaire à la fois pour les analyser mais aussi préserver leur qualité au fur et à mesure de leur traitement.

References

[1] Bhat, A., et al. (2019). A Machine Learning Approach for Identifying Compound Words from a Sanskrit Text. In P. Goyal (Ed.), Proceedings of the 6th International Sanskrit Computational Linguistics Symposium (pp. 45–51). Association for Computational Linguistics. https://aclanthology.org/W19-7504

[2] Bartolotta, A., et al. (2015). Homeric Greek Compound Project, University of Palermo.

[3] Bonami, O., & Pellegrini, M. (2022). Derivation predicting inflection: A quantitative study of the influence of derivational history on inflectional behavior in Latin. *Studies in Language, 46*, 753–792.

[4] Rousseau, N. (2016). Du syntagme au lexique: Sur la composition en grec ancien. *Les Belles Lettres*.

DÉCODAGE PARESSEUX: DÉCODAGE CONTRAINT POUR L'EXTRACTION D'INFORMATIONS

Arthur Hemmer[1,2], **Mickaël Coustaty**[2], **Nicola Bartolo**[1], **Jérôme Brachat**[1], **Jean-Marc Ogier**[2]
[1] Shift Technology, Paris, France
[2] L3i, La Rochelle, France

Aujourd'hui, l'extraction d'informations à partir de documents administratifs est principalement faite avec des LLMs et des modèles de langue comme BERT [1] et ses dérivés pour les documents comme par exemple LayoutLM [2] et LiLT [3]. Ces modèles atteignent de bons résultats en empilant le plus de paramètres et de données possible, ce qui entraîne cependant un coût computationnel et de complexité d'entraînement élevé. Typiquement, seule la prédiction la plus probable (top-1) est utilisée, malgré le fait que les modèles produisent des probabilités pour tous les labels.

Dans ce travail, nous explorons des prédictions alternatives à forte probabilité pour améliorer les prédictions issues de modèles existants. Cela est particulièrement pertinent dans les tâches de prédiction structurée, où les prédictions du modèle sont analysées et intégrées dans des structures prédéfinies. Ces structures permettent de définir des contraintes pour évaluer si une prédiction produite adhère à la structure attendue, ce qui peut ensuite être utilisé pour séquentiellement parcourir plusieurs prédictions à haute probabilité pour trouver une solution qui satisfait les contraintes.

Plus spécifiquement, nous combinons les modèles probabilistes avec des approches de décodage sous contraintes, dans le contexte de la classification de tokens pour l'extraction d'informations à partir de documents administratifs. Nous formulons des contraintes comme "le montant total doit être égal au montant payé en espèces moins la monnaie" et "le montant total doit être égal à la somme des lignes", et utilisons des méthodes de décodage qui cherchent des combinaisons de labels qui satisfont les contraintes en maximisant la probabilité totale donnée par le modèle.

Nous évaluons plusieurs approches existantes, et proposons également une méthode de décodage nommée Lazy-k, basé sur du A* avec expansion partiale [4]. Nos résultats démontrent que les approches de décodage sous contraintes peuvent significativement améliorer les prédictions des modèles, particulièrement lors de l'utilisation de modèles plus petits. De plus, l'approche Lazy-k permet une plus grande flexibilité entre le temps de décodage et le F1-score comparée à d'autres méthodes de décodage sous contraintes.

Références

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT : Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[2] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. Layoutlm : Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 1192–1200, New York, NY, USA, 2020. Association for Computing Machinery.

[3] Jiapeng Wang, Lianwen Jin, and Kai Ding. LiLT : A simple yet effective language-independent layout transformer for structured document understanding. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 7747–7757, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[4] Takayuki Yoshizumi, Teruhisa Miura, and Toru Ishida. A* with partial expansion for large branching factor problems. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, page 923–929. AAAI Press, 2000.

OÙ LA FRUGALITÉ REJOINT L'ÉTHIQUE : UTILISATION DE DONNÉES SYNTHÉTIQUES POUR LA RECONNAISSANCE D'ENTITÉS CLINIQUES

Nicolas Hiebel[1], Olivier Ferret[2], Karën Fort[3], Aurélie Névéol[1]
(1) Université Paris-Saclay, CNRS, LISN, 91400, Orsay, France
(2) Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
(3) Sorbonne Université / Université de Lorraine, CNRS, Inria, LORIA, 54506, Vandœuvre-lès-Nancy, France
[1]prenom.nom@lisn.upsaclay.fr, [2]olivier.ferret@cea.fr, [3]karen.fort@loria.fr

Contexte

Dans certains domaines du traitement automatique des langues (TAL), comme le domaine médical, le fait de ne pas pouvoir partager des données crée une contrainte de frugalité. Les corpus cliniques dont l'accès est relativement facile en français (E3C [1], CAS [2]) ne sont pas tout à fait représentatifs des documents confidentiels présents dans les hôpitaux. Le partage des connaissances au sein de la communauté scientifique est compliqué. Aucune reproductibilité n'est possible, tout comme les comparaisons avec d'autres méthodes ou données. Une piste pour s'affranchir de ce manque de données partageables est de générer des documents synthétiques similaires aux documents réels mais ne présentant pas d'informations sensibles. Cela pourrait permettre à des personnes ayant accès à un corpus protégé de générer un corpus librement distribué à partir du premier. En partageant la méthode de génération, il serait également possible de reproduire l'expérience sur d'autres données confidentielles. La mise à disposition des données générées donnerait alors à la communauté scientifique un terrain de test, de comparaison, de discussion et d'entraide dans la recherche en TAL biomédical, tout en préservant la confidentialité des données.

Cependant, l'utilité des données synthétiques générées n'est pas facile à évaluer, particulièrement dans le cas où le corpus dont s'inspirent les données synthétiques n'est pas annoté. Dans le cadre de l'évaluation de la qualité des données synthétiques, nous présentons ici un protocole pour mesurer la pertinence des données synthétiques afin d'entraîner des modèles de TAL.

Objectifs À la base de ce travail, nous générons des textes synthétiques à partir de cas cliniques réels selon plusieurs configurations et nous souhaitons mesurer leur qualité. L'objectif est d'obtenir des données synthétiques partageant suffisamment de caractéristiques avec les données réelles, de manière à ce qu'il soit aussi pertinent de travailler sur les données synthétiques que sur les données réelles. Dans le même temps, les données synthétiques doivent être suffisamment éloignées des données réelles pour limiter le risque de divulguer des informations sensibles présentes dans les données réelles. Dans cette étude, nous évaluons la pertinence de ces cas cliniques synthétiques en nous intéressant à leur utilité pour une tâche classique en TAL : la reconnaissance d'entités nommées (REN), ici dans le domaine clinique. Parallèlement, nous évaluons la proximité des données synthétiques avec les données d'origine à l'aide de recouvrement de ngrammes et analysons manuellement les types d'erreurs présentes dans les cas cliniques générés.

Méthodologie

Corpus Deux corpus biomédicaux interviennent dans cette étude. Le premier est le corpus multilingue librement disponible E3C. Nous sélectionnons ici uniquement les cas cliniques en français, sous-ensemble que nous appellerons E3C_{FR}. Ce corpus est utilisé pour générer les documents synthétiques. Le second est le corpus MERLOT [3], un corpus clinique de 500 documents dont l'accès est restreint. Ce corpus contient des annotations manuelles en entités. Ces annotations sont utilisées à la fois pour entraîner des modèles de reconnaissance d'entités cliniques et pour évaluer les modèles entraînés sur les autres corpus.

Génération Nous avons ici choisi d'explorer la capacité des modèles neuronaux auto-régressifs pré-entraînés, comme GPT2 [4] et BLOOM [5], à s'adapter au domaine médical et à générer des cas cliniques. Nous avons sélectionné un modèle français que nous appellerons LLF [1] et un modèle multilingue, en l'occurrence une des versions du modèle BLOOM [2]. Pour pouvoir comparer les résultats, nous avons choisi deux modèles de taille à peu près équivalente (environ 1 milliard de paramètres). Les modèles sont entraînés à générer des documents cliniques entiers, en

1. https://huggingface.co/asi/gpt-fr-cased-base
2. https://huggingface.co/bigscience/bloom-1b1

	Test					
	$\texttt{E3C}_{FR}$			**MERLOT**		
Training	**P**	**R**	**F**	**P**	**R**	**F**
$\texttt{E3C}_{FR}$	**89,5**	**91,0**	**90,2**	64,4	78,0	70,5
MERLOT	87,1	90,8	88,9	**85,2**	**85,8**	**85,5**
$\texttt{Bloom}_{E3C}$	87,6	87,9	87,8	63,1	74,9	68,5
$\texttt{LLF}_{E3C}$	87,6	87,2	87,4	64,5	76,4	70,0
$\texttt{Bloom}_{E3C+T}$	83,4	68,9	75,4	71,7	55,1	62,3
$\texttt{LLF}_{E3C+T}$	84,4	68,1	75,4	75,7	46,4	57,5

TABLEAU 1 – Résultats de la tâche de REN sur les corpus réels (P=précision, R=rappel, F=F-mesure).

indiquant seulement le début et la fin d'un document par des balises. À la génération, seule une balise de début de document est donnée en amorce. Deux configurations d'entraînement sont testées pour chaque modèle. Pour la première, les modèles sont entraînés sur les données d'$\texttt{E3C}_{FR}$ brutes. Pour la seconde, les données d'$\texttt{E3C}_{FR}$ sont d'abord annotées par les modèles REN entraînés sur MERLOT et les annotations sont ensuite intégrées aux textes sous forme de balises de début et de fin d'entités. Les modèles sont entraînés sur les textes contenant les annotations. Ainsi, les modèles apprennent à générer les annotations en même temps que le texte. Les corpus synthétiques générés seront nommés $\texttt{Bloom}_{E3C}$ et $\texttt{LLF}_{E3C}$ pour les versions entraînées sans annotation et $\texttt{Bloom}_{E3C+T}$ et $\texttt{LLF}_{E3C+T}$ pour les autres.

Filtrage Les textes générés avec des modèles auto-régressifs peuvent présenter des défauts, comme des répétitions de tokens, ou bien un manque de diversité entre les différentes générations. C'est pourquoi nous générons beaucoup plus de tokens que le nombre de tokens visé (pour obtenir pour chaque configuration autant de tokens que dans le jeu d'entraînement). Nous avons choisi ici de sélectionner des documents divers et d'éliminer les cas où la génération présente une erreur facilement repérable (tokens extrêmement longs, boucle de répétition de tokens…).

Trois niveaux d'évaluation Les textes générés sans annotation sont annotés de la même manière qu'$\texttt{E3C}_{FR}$ à l'aide des modèles REN entraînés sur MERLOT. Ensuite, de nouveaux modèles REN sont entraînés sur chaque corpus et tous les modèles REN sont testés sur les jeux de test de tous les corpus annotés obtenus.

Nous ajoutons à cela une mesure du pourcentage de recouvrement de ngrammes entre chaque corpus généré et le corpus d'origine $\texttt{E3C}_{FR}$ pour évaluer la proximité des corpus.

Enfin, nous avons réalisé une analyse manuelle de 15 documents pour chaque configuration (pour un total de 60 documents et 13 809 tokens) visant à évaluer la grammaticalité et la cohérence clinique des documents générés.

Résultats

Le tableau 1 présente les résultats de la reconnaissance d'entités cliniques sur les jeux de test des corpus réels. Ce tableau montre la pertinence des corpus générés pour notre tâche de reconnaissance d'entités cliniques, notamment avec les résultats sur le corpus MERLOT. Comme il a été annoté manuellement, contrairement aux autres corpus, nous considérons MERLOT comme une référence de haute qualité. Bien que les modèles entraînés sur $\texttt{E3C}_{FR}$ soient significativement moins performants que les modèles entraînés sur MERLOT, nous pouvons observer que les performances des modèles entraînés sur $\texttt{Bloom}_{E3C}$ et $\texttt{LLF}_{E3C}$ sont proches de celles des modèles entraînés sur $\texttt{E3C}_{FR}$. Ainsi, dans notre contexte, l'utilisation d'un corpus généré à partir d'un corpus réel est relativement équivalente à l'utilisation du corpus réel. Cela se confirme avec les résultats sur le corpus $\texttt{E3C}_{FR}$.

Le tableau 2 présente le recouvrement de ngrammes entre le corpus E3C et les corpus générés et un autre corpus réel de cas cliniques, CAS.

On constate que le corpus réel CAS présente le plus de 1-grammes, donc de vocabulaire, en commun avec le corpus E3C. En revanche, CAS est l'un des corpus comparés avec le moins de séquences longues (8-grammes) en commun avec E3C. Parmi les corpus générés, le corpus $\texttt{Bloom}_{E3C}$ est prometteur car il contient le plus grand nombre de 1-grammes en commun et le plus petit nombre de 8-grammes en commun. À l'inverse, le corpus $\texttt{LLF}_{E3C}$ possède le plus petit nombre de 1-grammes en commun tout en ayant le plus grand nombre de 8-grammes en commun, ce qui est moins souhaitable.

Conclusion

	Corpus	1gram	...	4gram	...	8gram
Synthétique	$\texttt{Bloom}_{E3C}$	0,16419		0,00368		0,00011
	$\texttt{Bloom}_{E3C+T}$	0,13887		0,00531		0,00020
	$\texttt{LLF}_{E3C}$	0,11740		0,00447		**0,00023**
	$\texttt{LLF}_{E3C+T}$	0,11935		0,00505		0,00013
Réel	CAS	**0,20373**		0,00899		0,00013

TABLEAU 2 – Recouvrement de ngrammes entre les corpus générés et E3C. Chaque ligne correspond à la comparaison entre le corpus et E3C. La comparaison entre CAS et E3C sert de baseline.

Nous présentons dans ce travail quatre modèles de génération de cas cliniques synthétiques en français et nous montrons qu'entraîner des modèles de reconnaissance d'entités cliniques sur ces textes synthétiques est pratiquement équivalent à entraîner des modèles sur les données réelles dont ils sont issus, sans pour autant les copier.

Références

[1] Bernardo Magnini, Begoña Altuna, Alberto Lavelli, Manuela Speranza, and Roberto Zanoli. The E3C Project : Collection and Annotation of a Multilingual Corpus of Clinical Cases. In Johanna Monti, Felice dell'Orletta, and Fabio Tamburini, editors, *Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020, Bologna, Italy, March 1-3, 2021*, volume 2769 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

[2] Natalia Grabar, Vincent Claveau, and Clément Dalloux. CAS : French corpus with clinical cases. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 122–128, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[3] Leonardo Campillos, Louise Deléger, Cyril Grouin, Thierry Hamon, Anne-Laure Ligozat, and Aurélie Névéol. A French clinical corpus with comprehensive semantic annotations : development of the Medical Entity and Relation LIMSI annOtated Text corpus (MERLOT). *Language Resources and Evaluation*, 52(2) :571–601, 2018.

[4] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

[5] Teven Le Scao et al. Bloom : A 176b-parameter open-access multilingual language model, 2022.

PRACTICAL CONSIDERATIONS FOR IMPLEMENTING SPARSE TENSOR MANIPULATION OVER ARBITRARY SEMIRINGS

Lucas Ondel Yang
Université Paris-Saclay, CNRS, LISN, 91400, Orsay, France

1 Motivation

A major challenge in speech-related applications is the possibility to represent linguistic constraints withing end-to-end processing pipeline. To represent such constraints and to structure the search space during inference, Weighted Finite State Transducers [1] have been a tool of choice for decades. However, their application is moder settings proves to be challenging: in a large neural-network dominated field WFST are not easily amenable to end-to-end training fashion. Difficulties to backpropagate the gradient as well as lack of parallel implementation for WFST-based algorithms prevent their integration. Recently, efforts have been made to overcome these barriers [2] and to provide practical implementation of major WFST algorithms which are differentiable (w.r.t. the weights of the WFST) and executable on massively parallel device. This approach relies on representing WFST as sparse tensors whose element belongs to a user-defined semiring. While preliminary results are encouraging, the implementation of such framework remains a difficult endeavor. We propose in this communication to discuss two challenges along with their practical solution.

2 Sum of sparse tensors rather than sparse tensors

Let be $T = (\Sigma, \Delta, Q, E, I, F, \alpha, \omega)$ a transducer over a semiring defined a usual with a set of input symbols Σ, a set of output symbols Δ, a set of states Q, a set of transition E, I a set of initial states, F a set of final states and α and ω are the weighting function assigning an initial and a final weight to states in I and F respectively. A transducer can be represented as a triplet $T = (\mathbf{M}, \boldsymbol{\alpha}, \boldsymbol{\omega})$ where $\mathbf{M} \in S^{|Q|\times|Q|\times|\Sigma|\times|\Delta|}$, $\boldsymbol{\alpha} \in S^{|Q|}$ and $\boldsymbol{\omega} \in S^{|Q|}$.

Under this formalism, many structured inference problems (including speech recognition) can be expressed as:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \{W[T_1(\boldsymbol{\theta}) \circ T_2]\} \tag{1}$$

where $T(\boldsymbol{\theta})$ is the output of the function of $\boldsymbol{\theta}$ interpreted as a task specific transducer and T_2 are user-defined rules limiting the search space of the problem, $\circ$ is the transducer composition operation and $W(T)$ is the weight of a transducer T, i.e. the $\oplus$-sum of all its paths. Using our algebraic definition, one can show that evaluating the argument of the $\arg\max$ function leads to the following a dynamic program

$$\mathbf{u}_{n+1}^\top = \mathbf{u}_n^\top \sum_{i,j\in\Sigma\times\Delta} \mathbf{M}_{1,ij} \odot \mathbf{M}_{2,ij} \tag{2}$$

where $\odot$ is the matrix Kronecker product. The $\odot$ operation has a work depth of $O(1)$ and therefore benefits highly from the parallel acceleration.

A direct implementation of 2 would encode the matrices $\mathbf{M}_{1,ij}$ and $\mathbf{M}_{1,ij}$ in a sparse format such as Compressed Sparse Row or similar. This approach is however highly suboptimal as it would necessitate the evaluate the sum[1] of label i, j prior to perform the vector-matrix multiplication.

We propose another approach which consists in encoding the *sum of the matrices* in a sparse format rather the just the matrices themselves. Concretely, this is achieved trivially by allowing the sparse tensor format to have duplicate coordinates along with an adequate processing while carrying the multiplication. Formally, this is simply delaying the summation by rewriting (2) as

$$\mathbf{u}_{n+1}^\top = \sum_{i,j\in\Sigma\times\Delta} \mathbf{u}_n^\top \mathbf{M}_{1,ij} \odot \mathbf{M}_{2,ij} \tag{3}$$

Using sparse array format to store sum of sparse arrays is a key ingredient to implement on-demand FST algorithm avoiding the bottleneck of materalizagin gigantic structure in memory.

[1] Summing sparse arrays is costly as the sparsity pattern of the resulting array is not known before hand.

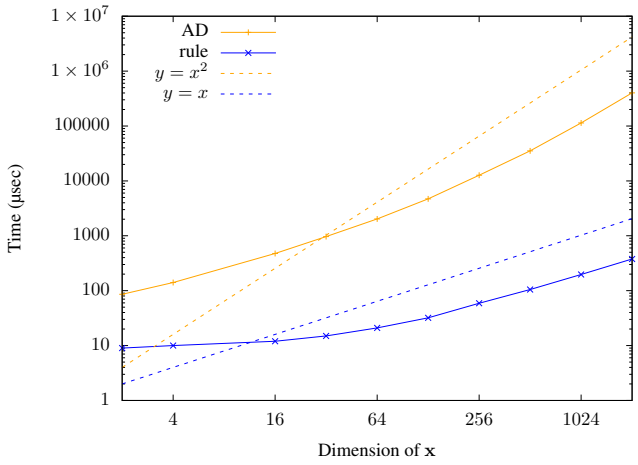


Figure 1: Execution time of evaluating the gradient of $\nabla_{\mathbf{x}} f(\mathbf{x})$ with the log-semiring by using Zygote [3] AD engine or by implementing manually the differentiation rule. Dashed lines represent quadratic and linear trends.

3 Differentiating through arbitrary semiring

The notion of derivatives is not defined for arbitrary semirings therefore we restrict our discussion to semiring $(S' \subseteq \mathbb{R}, \oplus, \otimes, \bar{0}, \bar{1})$ where $\oplus$ and $\otimes$ are binary functions differentiable with respect to both arguments. To keep the notation uncluttered we use the following notation

$$x \dot{\otimes}_l y = \frac{\partial x \otimes y}{\partial x} \qquad x \dot{\otimes}_r y = \frac{\partial x \otimes y}{\partial y} \tag{4}$$

$$x \dot{\oplus}_l y = \frac{\partial x \oplus y}{\partial x} \qquad x \dot{\oplus}_r y = \frac{\partial x \oplus y}{\partial y}. \tag{5}$$

Note that the functions $\dot{\otimes}_l, \dot{\otimes}_r, \dot{\oplus}_l$, and $\dot{\oplus}_r$ take semiring elements as input and return a value in $\mathbb{R}$.

We consider the problem of estimating the partial derivative of the dots product $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{y}, \mathbf{x}, \mathbf{y} \in M_d(S)$. Following the above notation and making use of the commutativity of $\oplus$, the derivative of the dot product in $M_d(S)$ can be expressed as

$$\frac{\partial f(\mathbf{x})}{\partial x_i} = \Big((x_i \otimes y_i) \dot{\oplus}_l v_i\Big)(x_i \dot{\otimes}_l y_i) \tag{6}$$

where

$$v_i = \Big(\bigoplus_{j<i} x_j \otimes y_j\Big) \oplus \Big(\bigoplus_{j>i} x_j \otimes y_j\Big). \tag{7}$$

From (6) it follows that the complexity to evaluate the gradient $\nabla_{\mathbf{x}} f$ is $\mathcal{O}(d^2)$. This contrasts with the gradient of the dot product in the field of scalar which has a linear complexity.

The complexity of the gradient of the dot product in $M_d(S)$ can be improved when there exists an isomorphism $\mu : (S', \oplus, \bar{0}) \mapsto (\mathbb{R}, +, 0)$ which allows us to express f as:

$$f(\mathbf{x}) = \mu^{-1}[\mu(\mathbf{x}^\top \mathbf{y})] \tag{8}$$

$$= \mu^{-1}[\mu(x_1 \otimes y_1) + \cdots + \mu(x_d \otimes y_d)]. \tag{9}$$

Then, the partial derivatives of f becomes

$$\frac{\partial f(\mathbf{x})}{\partial x_i} = \frac{\partial \mu^{-1}}{\partial \mu(\mathbf{x}^\top \mathbf{y})} \frac{\partial \mu(x_i \otimes y_i)}{\partial x_i}. \tag{10}$$

If the complexity of evaluating μ is constant, then the complexity of taking the gradient $\nabla_{\mathbf{x}} f$ is $\mathcal{O}(d)$: $\frac{\partial \mu^{-1}}{\partial \mu(\mathbf{x}^\top \mathbf{y})}$ has to be computed once and $\frac{\partial \mu(x_i \otimes y_i)}{\partial x_i}$ has to be evaluated for all i.

The difference of complexity between (6) and (10) has important practical implications: when using Automatic Differentiation (AD) to evaluate the gradient of $f(\mathbf{x})$, direct application of the chain rule leads to a computation analog to (6) with quadratic complexity whereas expert-based implementation can make use of more efficient computation as in (10) with linear complexity. This implies that the design of an efficient package of differentiable linear algebra over semirings necessitate to provide differentiation rules to help the AD system to evaluate derivatives in a reasonable times.

References

[1] Mehryar Mohri, Fernando Pereira, and Michael Riley. *Speech Recognition with Weighted Finite-State Transducers*, pages 559–584. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[2] Awni Y. Hannun, Vineel Pratap, Jacob Kahn, and Wei-Ning Hsu. Differentiable weighted finite-state transducers. *ArXiv*, abs/2010.01003, 2020.

[3] Michael Innes. Don't unroll adjoint: Differentiating ssa-form programs. *CoRR*, abs/1810.07951, 2018.

DÉTECTION D'ÉVÉNEMENTS À PARTIR DE PEU D'EXEMPLES PAR MÉTA-APPRENTISSAGE

24 janvier 2024

1 Introduction

La détection d'événements est une tâche d'extraction d'information visant à extraire des instances de types d'événements donnés à partir de textes [1]. Cette extraction consiste à identifier des déclencheurs d'événements, qui sont des groupes de mots indiquant explicitement la présence d'un événement dans une phrase. Par exemple, dans la phrase « *John D. Idol will* ***take over*** *as Chief Executive.* », un événement « Start-Position » est déclenché par le déclencheur « *take over* ». Les approches d'apprentissage supervisé pour la détection d'événements ont été largement étudiées ces dernières années, notamment les méthodes fondées sur des traits lexico-syntaxiques [2, 3], les réseaux neuronaux convolutifs [4], les réseaux neuronaux récurrents [5] et les modèles fondés sur les graphes [6, 7, 8]. Cependant, toutes ces approches reposent sur des ensembles de données annotées conséquents pour l'entraînement, généralement difficiles à obtenir et donc peu compatibles avec une vision d'un Traitement Automatique des Langues (TAL) frugal du point de vue des données annotées.

La détection d'événements à partir de peu d'exemples (*Few-Shot Event Detection*, FSED) a donc suscité un grand intérêt ces dernières années avec l'émergence de méthodes d'apprentissage à partir de peu de données, notamment via le méta-apprentissage [9], et le développement de modèles de langue pré-entraînés capables de transférer leurs connaissances linguistiques à de nouvelles tâches. Cette FSED a été mise en œuvre sous différentes formes : *identification d'événement*, qui détermine si un mot dans une phrase est un déclencheur selon un type d'événement [10], *classification d'événements*, dont l'objectif est de choisir le type d'événement associé à un déclencheur déjà identifié dans une phrase [11, 12], et *la détection d'événements*, qui réalise ces deux étapes conjointement [13, 14].

Ces efforts de recherche ont fait de la FSED une tâche d'annotation de séquences, qui se transforme en un problème de classification de mots traité à l'aide de réseaux prototypiques [9], qui sont particulièrement adaptés à l'apprentissage à partir de peu d'exemples. Dans ce contexte, un prototype est construit pour chaque type d'événement ainsi que pour la classe « non-événement » (aussi appelée classe nulle) à partir d'un encodeur de séquences, puis, en inférence, les mots des phrases à annoter sont étiquetés en fonction de leur similarité à ces prototypes. Les modèles de l'état de l'art utilisent en général des modèles de langue multicouche pour obtenir les représentations des mots. Ces modèles n'exploitent pas de façon explicite les couches cachées de ces encodeurs, bien que celles-ci contiennent également de l'information. Par ailleurs, l'hétérogénéité intrinsèque du prototype « non-événement » rend difficile la discrimination entre les mots déclencheurs et non déclencheurs fondée sur la similarité avec les prototypes.

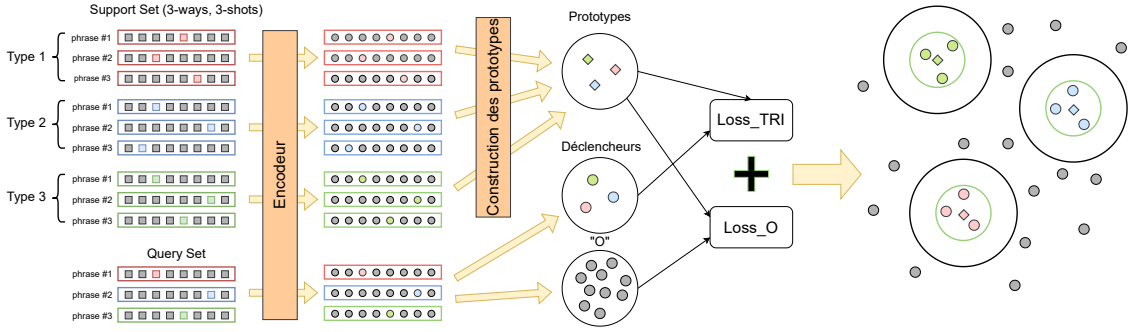
Dans ce travail, nous proposons une meilleure exploitation de ces modèles multicouche en étudiant l'importance de ces différentes couches et en évaluant différentes façons de les associer pour obtenir de meilleurs prototypes pour les événements. De plus, nous formulons la FSED comme un problème de détection d'exemples hors domaine [15], en considérant les mots de la classe nulle comme des exemples hors-domaine et en apprenant un seuil de similarité dynamique en deçà duquel ces exemples ne sont associés à aucune classe d'événement.

2 Méthode

Nous abordons la FSED par le biais d'un apprentissage épisodique à N-ways et k-shots [16] avec des réseaux prototypiques. À chaque épisode, nous considérons un sous-ensemble de phrases annotées appelé *support set* qui contient N types d'événements et k exemples annotés par type (k étant petit, typiquement entre 1 et 10). Un second ensemble, appelé *query set*, est utilisé pour faire des prédictions fondées sur les exemples annotés du *support set*. Chaque phrase peut contenir un ou plusieurs déclencheurs, associés chacun à un type d'événements. L'identification du type et de la position du déclencheur est effectuée en attribuant une étiquette à chaque mot, ce qui correspond à une classification multi-classe au niveau des mots, avec autant de classes que de types d'événements plus la classe nulle (étiquette « O ») pour les mots non-déclencheurs d'événements.

Nous construisons un prototype pour chaque classe à partir des exemples du *support set* en prenant la moyenne des représentations des k déclencheurs de cette classe. Ensuite, nous classons chaque mot du *query set* en fonction de sa similarité avec ces prototypes. Pendant l'apprentissage, ces similarités sont utilisées pour mettre à jour les poids du modèle via une fonction de coût.

FIGURE 1 – Vue d'ensemble du modèle.



Ce cadre prototypique est souvent décomposé en trois étapes, chacune jouant un rôle particulier dans le processus global de classification.

Un module d'encodage, qui se concentre sur la création de représentations vectorielles pour chaque exemple ou chaque classe, ces représentations jouant un rôle majeur dans la généralisation du modèle à de nouvelles classes. En effet, si les représentations vectorielles sont suffisamment riches en informations discriminantes et capturent efficacement les caractéristiques distinctives de chaque classe, alors le modèle sera en mesure de bien généraliser à de nouvelles classes.

Un module prototypique, qui concerne la manière de construire les prototypes à partir du *support set*, de combiner les informations de ces prototypes et de gérer les interactions entre eux.

Un module de prédiction, dédié au calcul des similarités entre les prototypes et les exemples du *query set* ainsi qu'à la manière dont les poids du modèle sont mis à jour lors de l'entraînement. Ce module intègre des aspects tels que le choix de la fonction de similarité ou la sélection d'une fonction de coût pour mettre à jour les poids de l'encodeur. L'objectif est de garantir que les prototypes et les exemples du *query set* sont correctement appariés, permettant ainsi au modèle de réaliser une classification précise malgré la faible quantité d'exemples disponibles.

Dans ce travail, nous présentons deux contributions, l'une s'appuyant sur le module d'encodage et l'autre sur le traitement particulier de la classe nulle. Une vue d'ensemble du modèle est donnée à la figure 1.

2.1 Enrichissement des représentations des événements

Cette première contribution concerne exclusivement le module d'encodage dans le cadre prototypique. Notre objectif ici est d'étudier différentes options pour exploiter de façon explicite les 12 couches de BERT [17] afin d'obtenir des représentations de mots plus riches pour la tâche de détection d'événements. Pour cela, nous explorons cinq configurations pour la sélection et la combinaison des couches : 1. **Average**, qui calcule la moyenne des couches sélectionnées, 2. **Max-pool**, qui effectue un max-pooling sur les couches sélectionnées, 3. **Concat**, qui réalise une concaténation des couches, 4. **Weighted**, qui effectue une moyenne pondérée avec des poids appris pendant l'apprentissage épisodique, et 5. **ATT**, qui construit une moyenne pondérée via un mécanisme d'attention.

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 [13] : **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 champs aléatoires conditionnels (CRF) [18] pour estimer les probabilités de transition entre les différentes étiquettes BIO.

La figure 2 montre d'abord que, quel que soit le modèle utilisé, toutes les modifications de l'encodeur, 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 [13], une référence de 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 prédiction).

2.2 Traitement spécifique de la classe nulle

Cette seconde contribution concerne les modules prototypique et de prédiction. Inspirés par les efforts de recherche sur la classification d'exemple hors-domaine avec peu d'exemples [19], nous évitons de construire le prototype « O » et proposons une approche fondée sur un seuillage dynamique adapté à chaque phrase en utilisant la fonction de répartition des similarités entre les mots et les prototypes.

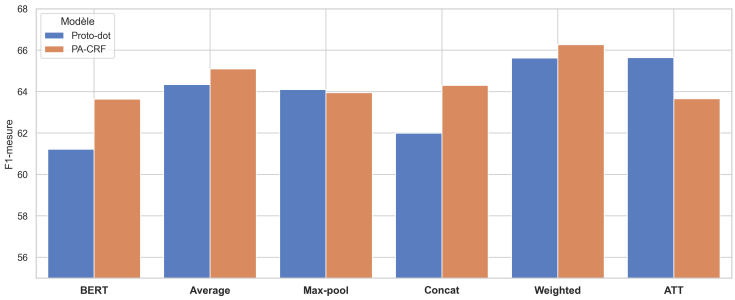


FIGURE 2 – F1-mesure pour les différents encodeurs.

Pour cela, nous proposons un apprentissage contrastif permettant de rapprocher les déclencheurs de leurs prototypes tout en écartant les non-déclencheurs de tous les prototypes. Pour l'apprentissage du modèle, nous adoptons une fonction de coût de type *hinge loss* comportant deux termes : **Loss-TRI**, qui rapproche le déclencheur de son prototype et **Loss-O**, qui éloigne les mots « O » de tous les prototypes.

En l'absence de prototype pour la classe nulle, nous devons nous fier à un seuil en dessous duquel le mot est considéré comme un non-déclencheur. Typiquement, dans des travaux tels que [19], un seuil global est défini en utilisant la distribution des valeurs de similarité sur un ensemble de validation. Cependant, dans notre cas, nous avons observé empiriquement que les distributions des valeurs de similarité entre un déclencheur et les prototypes varient trop d'une phrase à l'autre, ce qui rend impraticable l'utilisation d'un seuil global.

Pour résoudre ce problème, nous proposons de rechercher la probabilité correspondant au seuil optimal en utilisant la fonction de répartition sur les valeurs maximales de similarité. Ceci nous permet d'obtenir un seuil dynamique spécifique à la phrase considérée. Plus précisément, étant donné que les similarités des déclencheurs sont plus élevées que celles des mots « O », nous supposons que, pour une phrase donnée, les similarités des déclencheurs ne seront présentes qu'au-dessus d'un certain quantile (assez élevé) dans la distribution des similarités. Nous supposons également que ce quantile est assez stable, même s'il ne correspond pas à la même valeur de similarité d'une phrase à l'autre. En pratique, pour une phrase donnée du *query set*, nous sélectionnons la phrase la plus similaire dans le *support set*. Puis, nous faisons varier le seuil entre les similarités minimum et maximum et adoptons celui maximisant la F1-mesure sur la phrase sélectionnée. Ensuite, nous déterminons la probabilité correspondant à ce seuil en utilisant la fonction de répartition sur les valeurs de similarité. Enfin, nous déterminons le seuil optimal pour la phrase du *query set* à partir de sa fonction de répartition et de la probabilité déterminée précédemment. Toutefois, comme les probabilités directement calculées à partir de la fonction de répartition dépendent du nombre de mots dans les phrases, nous interpolons linéairement la fonction de répartition sur un plus grand nombre de points avant d'estimer les probabilités, ce qui nous permet de donner artificiellement à toutes les phrases la même longueur.

Enfin, nous effectuons un filtrage supplémentaire en fonction des catégories morphosyntaxiques (*PoS tags*), en ne conservant que les étiquettes les plus couramment associées aux déclencheurs d'événements.

3 Résultats

Les principaux résultats sont consignés dans le tableau 1. Nous comparons notre approche à trois autres modèles de l'état de l'art dans la configuration 5-ways 5-shots. **PA-CRF** [13] est un modèle de l'état de l'art construisant un prototype pour la classe nulle et estimant les probabilités de transition entre les étiquettes BIO avec l'utilisation de couches CRF. **HCL-TAT** [20] est également un modèle sans prototype pour la classe nulle utilisant un seuil de décision égal à la moyenne des similarités pendant un épisode. Nous comparons ces méthodes à un modèle prototypique de base qui construit un prototype pour la classe nulle, utilise l'entropie croisée comme fonction de coût et un encodeur BERT standard (**PROTO**). **FS-Causal** [10] est un modèle ajoutant une prise en compte explicite des relations de causalité entre les déclencheurs et leur contexte pour résoudre le problème dit de la malédiction des déclencheurs (*trigger curse*). Comme les résultats rapportés pour **FS-Causal** ne sont évalués que classe par classe, ils correspondent à une configuration 1-way 5-shots.

Notre méthode établit une nouvelle performance de l'état de l'art avec une augmentation moyenne de 10 points de la F1-mesure pour les trois jeux de données considérés (cf. tableau 1). Les analyses suggèrent que l'encodeur *Weighted* et l'apprentissage contrastif, combinés à notre nouvelle formulation, jouent un rôle important dans la performance globale du modèle. Plus spécifiquement, nous pouvons noter que la fonction contrastive contribue fortement à diminuer

TABLE 1 – Performance de détection d'événement sur trois jeux de données, en moyenne et écart-type de la micro F1-mesure sur 5 essais. † indique les résultats issus de l'article original. * indique que la différence entre le meilleur modèle (**en gras**) et le deuxième (souligné) est statistiquement significative.

	Modèle	ACE 2005	MAVEN	FewEvent
5-ways, 5-shots	PROTO	49,2 ± 1,2	51,6 ± 1,4	53,6 ± 0,7
	PA-CRF [13]	64,0 ± 0,6	65,2 ± 0,3	65,3 ± 2,0
	HCL-TAT† [20]	–	–	66,9 ± 0,7
	Notre modèle	**74,0*** ± **1,1**	76,9 ± 1,1	**79,6*** ± **4,2**
	– PoS tags	72,2 ± 2,2	**77,5*** ± **0,8**	77,9 ± 3,9
	– contrastive	66,5 ± 5,7	63,1 ± 12,6	75,9 ± 5,4
	– weighted	59,2 ± 3,6	50,0 ± 2,3	70,9 ± 2,7
1w,5s	FS-Causal† [10]	76,9 ± 1,4	55,0 ± 0,4	–
	OUTFIT	80,9 ± 2,9	81,1 ± 1,1	79,1 ± 2,1

la variance des résultats. Nous pensons également que cette fonction, combinée à notre stratégie de recherche de seuil, contribue à la forte différence de performance avec HCL-TAT alors que nos problématiques sont initialement proches. Comme nos expériences préliminaires l'ont suggéré, le filtrage des déclencheurs candidats en fonction de leurs catégories morphosyntaxiques permet d'augmenter les performances de quelques points pour deux jeux de données. Toutefois, la condition sans ce filtrage, qui est la plus très directement comparable aux modèles de l'état de l'art, montre que celui-ci n'est pas le facteur principal des améliorations obtenues.

Références

[1] Thien Huu Nguyen and Ralph Grishman. 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*, pages 365–371, Beijing, China, July 2015. Association for Computational Linguistics.

[2] Qi Li, Heng Ji, and Liang Huang. Joint event extraction via structured prediction with global features. In *ACL*, pages 73–82, Sofia, Bulgaria, August 2013.

[3] Shasha Liao and Ralph" Grishman. Acquiring topic features to improve event extraction : in pre-selected and balanced collections. In *RANLP*. Association for Computational Linguistics, 2011.

[4] Thien Huu Nguyen and Ralph Grishman. Event Detection and Domain Adaptation with Convolutional Neural Networks. In *ACL-IJCNLP*, pages 365–371, Beijing, China, 2015.

[5] Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. Joint Event Extraction via Recurrent Neural Networks. In *NAACL-HLT*, pages 300–309, San Diego, California, 2016.

[6] Xiao Liu, Zhunchen Luo, and Heyan Huang. Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation. In *EMNLP*, pages 1247–1256, 2018.

[7] Thien Huu Nguyen and Ralph Grishman. Graph Convolutional Networks with Argument-Aware Pooling for Event Detection. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[8] Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. Event Detection with Multi-Order Graph Convolution and Aggregated Attention. In *EMNLP-IJCNLP*, pages 5766–5770, 2019.

[9] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[10] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Honey or Poison ? Solving the Trigger Curse in Few-shot Event Detection via Causal Intervention. *arXiv :2109.05747*, 2021.

[11] Shumin Deng, Ningyu Zhang, Jiaojian Kang, Yichi Zhang, Wei Zhang, and Huajun Chen. Meta-Learning with Dynamic-Memory-Based Prototypical Network for Few-Shot Event Detection. In *WSDM*, pages 151–159, Houston, TX, USA, January 2020.

[12] Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. Learning Prototype Representations Across Few-Shot Tasks for Event Detection. In *EMNLP*, pages 5270–5277, 2021.

[13] Xin Cong, Shiyao Cui, Bowen Yu, Tingwen Liu, Wang Yubin, and Bin Wang. Few-Shot Event Detection with Prototypical Amortized Conditional Random Field. In *Findings of ACL-IJCNLP*, pages 28–40, Online, August 2021.

[14] Aboubacar Tuo, Romaric Besançon, Olivier Ferret, and Julien Tourille. Better Exploiting BERT for Few-Shot Event Detection. In *NLDB*, page 291–298, Berlin, Heidelberg, 2022. Springer-Verlag.

[15] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 7 :1443–1471, 07 2001.

[16] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies (NAACL-HLT 2019)*, pages 4171–4186, 2019.

[18] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, pages 282–289, San Francisco, CA, USA, 2001.

[19] Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. Out-of-Domain Detection for Low-Resource Text Classification Tasks. In *EMNLP-IJCNLP*, pages 3566–3572. Association for Computational Linguistics, November 2019.

[20] Ruihan Zhang, Wei Wei, Xian-Ling Mao, Rui Fang, and Dangyang Chen. HCL-TAT : A Hybrid Contrastive Learning Method for Few-shot Event Detection with Task-Adaptive Threshold. In *Findings of the Association for Computational Linguistics : EMNLP*, pages 1808–1819. Association for Computational Linguistics, 2022.

DE L'INFLUENCE DES REPRÉSENTATIONS DES MOTS POUR LA DÉTECTION D'ANOMALIES TEXTUELLES DANS UN CADRE *k-classes-out*

Alicia Breidenstein[1] and Matthieu Labeau[1]

[1]LTCI, Télécom-Paris, Institut Polytechnique de Paris, France
{alicia.breidenstein, matthieu.labeau}@telecom-paris.fr

1 Introduction

La détection d'anomalies consiste à détecter ce qui s'écarte d'une notion de *normalité*, qui est définie par les données disponibles et est supposée être clairement délimitée [1], tandis que les anomalies sont en dehors de cette limite : en pratique, la difficulté principale est qu'il n'est en général pas possible de caractériser les anomalies. Il existe aujourd'hui de nombreuses méthodes qui utilisent l'apprentissage automatique pour la détection d'anomalies dans du langage naturel : cependant, dans la littérature, la plupart des travaux utilisent la détection d'anomalies dans un contexte de tâche de classification, pour retirer des exemples *hors-domaine* qui ne correspondent à aucune des classes prédéfinies. Néanmoins, ce cadre est relativement spécifique, et nous choisissons ici de nous intéresser à un cadre plus général de détection d'anomalies textuelles.

Les travaux plus généraux sont néanmoins plus rares, et semblent avoir été initialement inspirés par les méthodes non-supervisées de modélisation thématique. Dans ce cadre, la tâche de détection d'anomalies est habituellement construite à partir de données de classification, suivant la configuration *k-classes out* : chaque classe est successivement considérée comme étant la classe normale et les autres classes sont vues comme des anomalies. Les deux principales méthodes de l'état de l'art sont CVDD [2] et DATE [3]. CVDD (Context Vector Data Decription) utilise des représentations statiques pré-entraînées et apprend, à l'aide d'un mécanisme d'auto-attention, à représenter conjointement les documents ainsi qu'un ensemble réduit de "vecteurs de contexte". Ces vecteurs servent à représenter les différents modes de normalité des documents. DATE (Detecting Anomalies in Text using ELECTRA) entraîne un modèle ELECTRA [4] à détecter les éléments d'un document qui ont été modifiés par un générateur séparé. L'agrégation au niveau d'un document des probabilités de remplacement de ses mots permet alors de détecter les anomalies. [3] compare ces deux méthodes en utilisant deux jeux de données de classification 20 Newsgroups et AG News. La plupart des expériences sur ces méthodes ont été menées en les alimentant avec des données normales (cadre semi-supervisé), mais l'article implémente aussi un cadre non-supervisé dans lequel une part d'anomalies est introduite dans les données d'entraînement.

Notre objectif est de revisiter ces deux méthodes pour obtenir une image plus complète des facteurs pertinents pour la détection d'anomalies textuelles. Une première différence essentielle, non abordée dans la littérature, est que CVDD utilise des représentations pré-apprises de GloVe [5] et FastText [6], alors que DATE est entraîné seulement sur les données disponibles pour la détection d'anomalies. Nous y remédions en conduisant des expériences avec des représentations statiques obtenues uniquement avec les données d'entraînement spécifiques à la détection d'anomalie. Pour que notre comparaison soit équitable, nous comparons les méthodes dans les cadres semi-supervisé et non-supervisé, ce dernier se rapprochant plus d'un cas d'usage réaliste de la détection d'anomalies. Enfin, nous réalisons des expériences sur RNCP [7], un jeu de données diversifié et avec des textes courts. Nous comparons ces méthodes à deux méthodes simples de référence : nous utilisons tout d'abord OCSVM (One-class Support Vector Machine) [8], avec la même configuration que celle utilisée dans les deux articles. Nous comparons aussi ces méthodes avec l'utilisation du LSA (Latent Semantic Analysis), qui renvoie une approximation linéaire de rang peu élevé des représentations des documents : on peut utiliser la qualité des exemples reconstruits pour détecter les anomalies, sans se reposer sur des représentations intermédiaires. La Table 1 résume la portée de nos expériences et illustre les manques qu'elles comblent.

2 Expériences

Les méthodes CVDD et OC-SVM nécessitent des modèles de représentations pré-entraînées pour convertir les mots en vecteurs. Dans [2], ces représentations sont obtenues en utilisant de grandes quantités de données. Le modèle de détection d'anomalie a donc indirectement accès, durant l'entraînement, à des données bien plus générales que celles qu'il doit considérer comme normales. Au contraire, les autres modèles (DATE et LSA) ont accès uniquement

	Cadre		**Représentations**
Entraînement	Non-supervisé	Semi-supervisé	Données spécifiques uniquement
CVDD	•	X	•
OCSVM	•	X	•
DATE	X	X	X
LSA	X	•	X

TABLE 1 – Expériences possibles avec les deux cadres et avec des représentations entraînées uniquement sur les données spécifiques à la détection d'anomalies textuelles. Les points indiquent les expériences qui n'ont pas encore été conduites dans la littérature.

	AGNews			**20Ng**			**RNCP**		
	AUC	AUPR-i	AUPR-o	AUC	AUPR-i	AUPR-o	AUC	AUPR-i	AUPR-o
LSA	81.1	62.1	92.1	61.4	27.3	85.6	56.8	12.3	91.9
OC-SVM + FT_{Large}	81.7	67.6	90.2	66.4	38.0	86.5	56.2	12.6	91.7
OC-SVM + $\text{R}_{Specifique}$	**89.8**	**75.6**	**95.9**	**81.4**	*44.3*	**94.4**	**63.7**	**14.5**	**93.2**
CVDD + FT_{Large}	87.1	71.5	94.3	68.0	42.5	86.4	56.6	12.8	91.5
CVDD + $\text{R}_{Specifique}$	86.2	70.0	94.1	70.4	**45.3**	88.2	58.3	12.8	91.8
DATE	*88.5*	*73.7*	*95.2*	*70.9*	41.8	*89.8*	*59.4*	*12.9*	*92.4*

TABLE 2 – AUCs des expériences de détection d'anomalies sur tous les jeux de données, avec tous les modèles. Pour OC-SVM et CVDD, nous affichons les meilleurs résultats sur tous les hyperparamètres sur FT_{Large} et nos propres représentations $\text{R}_{Specifique}$.

aux données spécifiques à la tâche de détection d'anomalies. Pour mesurer l'impact des données d'entraînement et en particulier pour tenter de comprendre dans quelle dimension les meilleurs résultats de DATE dans [3] sont dus à l'utilisation d'un modèle plus complexe ou à un entraînement restreint aux données spécifiques, nous choisissons d'utiliser plusieurs types de plongements lexicaux pour CVDD et OCSVM.

Suivant [2], nous utilisons les plongements d'un modèle FastText (FT_{Large}) pré-entraîné sur les articles Wikipedia en anglais (ou français pour RNCP) pour assurer une bonne comparaison. Ensuite, nous apprenons deux types de plongements : les premiers entraînés sur la totalité du jeu de données spécifique, et les autres entraînés uniquement sur la classe normale de ces données. Pour éviter d'avoir des représentations apprises par un modèle de prédiction comme FastText sur des jeux de données trop petits, nous utilisons aussi une alternative traditionnelle en TAL, la matrice PPMI (Positive Pointwise-Mutual Information) [9], que nous réduisons à la dimension appropriée en utilisant une décomposition en valeurs singulières (SVD). Étant donnée la taille réduite de nos jeux de données, nous nous limitons à des plongements statiques.

Pour évaluer et comparer les différentes méthodes, nous utilisons l'AUROC ou AUC (Area Under Receiver Operating Curve), qui est largement employée dans la littérature sur la détection d'anomalies. Elle permet de mesurer les performances d'un classifieur binaire en calculant l'aire sous la courbe ROC, obtenue en traçant la courbe du taux de vrais positifs par rapport au taux de faux positifs. Elle permet donc de couvrir la plage de valeurs prises par le seuil entre normalité et anomalies sur les différentes valeurs possibles prises par le score d'anomalie. Nous comparons aussi les performances des différents modèles en utilisant l'AUPR (Area Under Precision Recall curve), qui est moins commune. Elle permet de mesurer les performances sur un jeu de données déséquilibré, ce qui est important pour la détection d'anomalies où la proportion d'anomalies peut être très basse, alors que c'est leur détection qui nous importe. Même si ce n'est pas le cas dans notre configuration expérimentale k-classes-out, nous utilisons cette mesure pour une analyse plus complète.

3 Résultats et discussion

La Table 2 présente les meilleurs résultats obtenus pour chaque modèle. Pour CVDD et OCSVM, nous présentons les résultats obtenus sur nos plongements avec des données spécifiques et ceux obtenus avec des représentations extérieures séparément. LSA donne les moins bons résultats, mais nous montre les performances que peut atteindre un simple modèle linéaire, comme un indicateur de la complexité de la tâche sur chaque jeu de données. Avec des plongements appris sur le jeu de données spécifique, OC-SVM surpasse CVDD sur tous les jeux de données. Il atteint de meilleurs résultats que DATE, en particulier sur 20 Newsgroups et RNCP, tout en étant nettement plus simple. La

Table 2, inclut des valeurs d'AUPR-i et d'AUPR-o, qui sont les valeurs d'AUPR calculées respectivement pour les classes normales (inliers) et les anomalies (outliers). Pour cette mesure, les performance d'un classifieur aléatoire correspondent au nombre d'exemples positifs divisé par la taille du jeu de donnée d'évaluation. Les résultats varient donc d'un jeu de données à l'autre, ne dépendant pas uniquement des performances du modèle, mais variant aussi avec le nombre de classes et leur taille. Dans un cadre non supervisé, nous avons pu vérifier que les performances des modèles décroissent lorsque le nombre d'anomalies dans le jeu de données d'entraînement augmente. Dans ce cadre, les performances des modèles les uns par rapport aux autres restent globalement les mêmes que dans un cadre semi-supervisé.

La Table 3 présente les temps de calculs moyens d'exécution des différents modèles. Les calculs ont été exécutés sur un cluster de calcul et le temps d'exécution dépend donc aussi des expérimentations d'autres utilisateurs en cours au moment de l'exécution. Néanmoins, lissés sur un grand nombre d'expériences, ces résultats permettent d'avoir un ordre de grandeur des temps d'exécution. CVDD et DATE ont été exécutés sur un cœur GPU alors que OC-SVM et LSA ont utilisé un cœur CPU. Malgré l'utilisation d'un cœur GPU, DATE a presque toujours les temps d'exécution les plus longs et OCSVM a des temps d'exécution raisonnables, étant donné son utilisation d'un CPU.

	AGNews	20Ng	RNCP
LSA	813.5	190.8	272.9
OC-SVM	1 149.0	134.1	671.2
CVDD	756.9	107.5	470.6
DATE	2 021.3	278.9	390.4

TABLE 3 – Temps de calcul moyen de chacun des modèles sur les trois jeux de données en secondes.

Nous avons donc comparé de façon équitable différentes méthodes existantes de détection d'anomalies textuelles dans une configuration k-classes-out et montré qu'entraîner les modèles uniquement sur les données spécifiques à la détection d'anomalies peut donner de meilleurs résultats. Cela permet à des méthodes simples et peu coûteuses, comme OC-SVM, d'obtenir de très bons résultats, comparables ou meilleurs que les modèles de l'état de l'art basés sur des architectures neuronales profondes, avec seulement les données disponibles. Ces résultats nous semblent aussi indicateurs d'un potentiel des méthodes d'adaptation des modèles pré-entraînés pour la détection d'anomalies.

Références

[1] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

[2] Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *ACL*, 2019.

[3] Andrei Manolache, Florin Brad, and Elena Burceanu. Date : Detecting anomalies in text via self-supervision of transformers. In *NAACL-HLT*, Online, 2021. Association for Computational Linguistics.

[4] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra : Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020.

[5] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe : Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv :1607.04606*, 2016.

[7] Mira Ait-Saada and Mohamed Nadif. Unsupervised anomaly detection in multi-topic short-text corpora. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1392–1403, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

[8] Bernhard Schölkopf, John Platt, John Shawe-Taylor, Alexander Smola, and Robert Williamson. Estimating support of a high-dimensional distribution. *Neural Computation*, 13 :1443–1471, 07 2001.

[9] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1) :22–29, 1990.