La robustesse de la traduction neuronale : les systèmes de traduction automatique neuronale à l' épreuve de la reproductibilité de l'expérience

Guillaume Wisniewski, Lichao Zhu, Jean-Baptiste Younès, Nicolas Ballier

La robustesse de la traduction neuronale : les systèmes de traduction automatique neuronale à l'épreuve de la reproductibilité de l'expérience

Guillaume Wisniewski Lichao Zhu
Jean-Baptiste Yunès Nicolas Ballier

{guillaume.wisniewski,lichao.zhu,jean.baptiste.yunes,nicolas.ballier}@u-paris.fr

1 Introduction et problématique

Il existe aujourd'hui de nombreuses implémentations de modèles de traduction automatique neuronale (TAN), chacune implémentant différents modèles de traduction et proposant de très nombreux paramètres allant du choix de l'architecture (nombres de couches cachées, taille des représentations, ...) au paramètre de l'algorithme d'apprentissage (choix du pas d'apprentissage, de la méthode d'optimisation, ...). Ces implémentations dépendent également de nombreuses bibliothèques (typiquement `pytorch`, `tensorflow` ou `cuda`) qui évoluent sans cesse et dont le comportement peut parfois changer significativement d'une version à l'autre.

La multiplication de ces paramètres et le nombre d'implémentations disponibles, s'ils traduisent le dynamisme de la recherche en TAN et les différents tâtonnements des chercheurs, soulèvent plusieurs problèmes et limitent la robustesse des expériences : le grand nombre de paramètres complique la description précise des expériences qui ont été réalisées, puisqu'il est difficile de savoir quels sont les paramètres ayant véritablement une influence sur le résultat d'une expérience et dont la valeur doit donc être documentée. Elle limite également l'accès à l'ingénierie de la traduction automatique, les non-spécialistes (typiquement, des traducteurs professionnels) pouvant être désemparés devant le nombre de réglages à effectuer sans véritablement savoir le rôle de chaque paramètre ni l'impact que celui-ci aura sur la qualité des traductions obtenues.

L'objectif de ce travail préliminaire est d'apporter un premier élément de réponse à ces problèmes, en comparant la qualité des traductions obtenues par différents systèmes de traduction automatique afin d'évaluer la robustesse de celles-ci au changement d'implémentation et, de ce fait, à certains choix techniques et à certains paramètres (dont la valeur n'est pas la même dans les différentes implémentations). Pour cela, nous proposons de comparer les performances obtenues par trois systèmes de traduction de l'état de l'art aussi bien à l'aide de métriques automatiques que de manière plus qualitative.

2 Expériences

Conditions expérimentales Pour comparer la robustesse de l'apprentissage d'un système de traduction neuronale, nous considérons trois implémentations d'une architecture `Transformer` aujourd'hui au cœur de tous les systèmes de traduction de l'état de l'art : `JoeyNMT` [2], `OpenNMT` [1] et `Nematus` [10]. L'expérience que nous proposons au vu de la problématique décrite dans la section précédente consiste à comparer la qualité des traductions obtenues en utilisant une version « sur étagère » de ces implémentations, c'est-à-dire sans régler aucun paramètre, exceptés ceux permettant de décrire le modèle de traduction utilisé. Nous avons utilisé la même architecture que celle utilisée dans le papier introduisant le modèle `Transformer` à savoir : un décodeur et un encodeur composés chacun de 6 couches avec 8 têtes d'attention, une représentation des unités lexicales sur 512 dimensions et une couche *feed-forward* de dimension 2 048. Les trois logiciels que nous avons considérés offrent la possibilité de modifier de nombreux paramètres en plus des 8 que nous venons de décrire. Par exemple, le fichier de configuration « de base » de `JoeyNMT` permet de spécifier 107 paramètres, une quinzaine correspondant à la définition des entrées/sorties (typiquement, nom des fichiers, des modèles, paramètres du tokenizer, fréquence du calcul de l'erreur en validation, ...), une quinzaine décrivant l'architecture `Transformer` à proprement parler et les autres correspondant aux paramètres de l'algorithme d'optimisation (il y a notamment une dizaine de paramètres permettant de configurer le pas d'apprentissage et le *dropout*) ou les paramètres du décodeur (typiquement la taille du faisceau).

Pour cette étude pilote, nous considérons un corpus de traduction de l'anglais vers le français issues du corpus `TED2020` [7], qui est constitué par les transcriptions des conférences « TED Talks » et des traductions de celles-ci par des volontaires de

	BLEU	CHRF2
JOEYNMT		
run 1	41,6	64,5
run 2	41,1	64,4
OPENNMT		
run 1	37,5	60,7
NEMATUS		
run 1	43,9	65,8
run 2	44,3	65,9

Table 1: Évaluation sur notre ensemble de test `TedTalk` de la qualité des traductions obtenues par plusieurs implémentations d'un modèle `Transformer`

ce projet [8]. Ce corpus est divisé en un ensemble d'apprentissage (395 849 phrases pour 8 millions de mots), de validation (2 000 phrases) et de test (2 000 phrases également). Toutes les données ont été segmentées en unités sous-lexicales à l'aide de `SentencePiece` [3] (pour l'entraînement de `JoeyNMT` et `OpenNMT`) et de `Subword-NMT` [9] (pour l'entraînement de `Nematus`). Nous avons choisi un vocabulaire de 32 000 tokens comme cela se fait habituellement en traduction automatique.

Résultats Nous avons reporté à la Table 1, les scores BLEU obtenus par les différents systèmes sur notre corpus de test, ainsi que les scores CHRF2 [5]. Ce dernier score correspond à score F_1 calculé sur les 6-grams de caractères. Utiliser une métrique au niveau des caractères permet de réaliser une évaluation qui est (en grande partie) indépendante de la segmentation en mot. Ces deux scores ont été calculés en utilisant SACREBLEU [6] sur les hypothèses (et les références !) segmentées en unités sous-lexicales. L'impact de la segmentation sur le calcul du score BLEU a été précisément documenté par [4] et la comparaison des scores BLEU obtenus par deux systèmes différents doit toujours être fait avec précaution.

Nous avons réalisé, pour JOEYNMT et NEMATUS, deux entraînements afin de mesurer l'impact du caractère aléatoire de la méthode d'optimisation et de pouvoir comparer la variabilité *intra-système* (différence de performance entre deux apprentissages avec une même implémentation) à la variabilité *inter-système* (différence de performance obtenue par deux systèmes de traduction différents). Notons que, étant donné les différents paramètres qui sont choisis aléatoirement lors de l'apprentissage (choix des neurones qui sont ignorés à cause du *dropout*, constitution des batch, ...) deux apprentissage d'un même système peuvent aboutir à des paramètres (et donc des performances) différents, même si, pour assurer la reproductbilité des expériences les systèmes fixent généralement la graine du générateur de nombre aléatoire. C'est notamment le cas pour JOEYNMT : pour mesurer la variabilité intra-système, nous avons du explicitement fixer la graine à des valeurs différentes.

Les résultats de la table 1 montrent que le choix d'une implémentation n'est pas anodin : les scores obtenus par présentent une variabilité forte d'une implémentation à l'autre. Il faut toutefois noter que ces scores ont été obtenus en utilisant les valeurs par défaut de nombreux paramètres et qu'il est probable qu'un réglage plus fin permette de réduire l'écart entre les systèmes. La comparaison des performances obtenues sur les deux entraînements de JOEYNMT montre que les différences de performances observées entre les différents systèmes sont bien dues soit à des différences d'implémentation soit au choix de certaines paramètres et non au caractère aléatoire de l'apprentissage.

Pour obtenir une image plus précise des différences entre les traductions obtenues par les différents systèmes, nous avons également calculé la distance d'édition (au niveau des caractères, pour ne pas dépendre de la segmentation en mots) entre les hypothèses des différents systèmes. La distance d'édition est une mesure de similarité (de « différence » pour être plus précis) entre chaînes de caractères qui peut être interprétée comme le plus petit nombre de caractères à modifier pour transformer une chaîne en une autre. L'objectif de cette deuxième série de mesures est de déterminer si les différences entre les scores BLEU reportées ci-dessus ont un véritable impact sur sur la qualité de la traduction. Nous avons représenté à la figure 1 la distribution des distances d'édition entre les hypothèses prédites par OPENNMT et les deux apprentissages de JOEYNMT. Cette figure montre que, à quelques exceptions, les hypothèses générées ne sont pas trop différentes.

Remerciements Nous remercions le CNRS/TGIR HUMA-NUM et le Centre de calcul IN2P3 (Lyon - France) pour la fourniture des ressources informatiques et de traitement d'une partie des données.

References

[1] Guillaume Klein et al. "OpenNMT: Open-Source Toolkit for Neural Machine Translation". In: *Proceedings of ACL 2017, System Demonstrations*. Vancouver, Canada: Association for Computational Linguistics, July 2017, pp. 67–72. URL: `https://www.aclweb.org/anthology/P17-4012`.

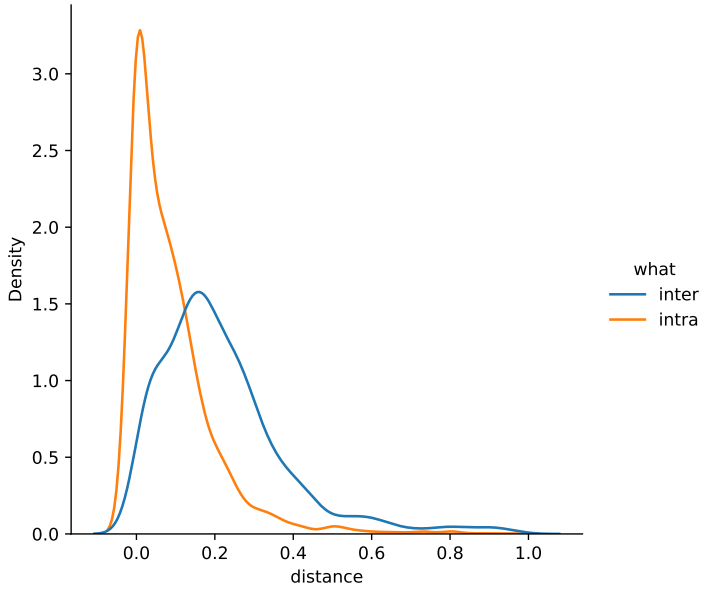

Figure 1: Distribution des distances d'édition intra-système (entre les deux entraînements de JoeyNMT) et inter-système (entre les prédictions de JoeyNMT et de openNMT

[2] Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. "Joey NMT: A Minimalist NMT Toolkit for Novices". 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): System Demonstrations*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 109–114. DOI: 10.18653/v1/D19-3019. URL: https://aclanthology.org/D19-3019.

[3] Taku Kudo and John Richardson. "SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 66–71. DOI: 10.18653/v1/D18-2012. URL: https://aclanthology.org/D18-2012.

[4] Benjamin Marie. *Science Left Behind*. 2022. URL: https://medium.com/@bnjmn_marie/science-left-behind-ca0a58231c20.

[5] Maja Popović. "chrF: character n-gram F-score for automatic MT evaluation". In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 392–395. DOI: 10.18653/v1/W15-3049. URL: https://aclanthology.org/W15-3049.

[6] Matt Post. "A Call for Clarity in Reporting BLEU Scores". In: *Proceedings of the Third Conference on Machine Translation: Research Papers*. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 186–191. DOI: 10.18653/v1/W18-6319. URL: https://aclanthology.org/W18-6319.

[7] Nils Reimers and Iryna Gurevych. "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2020. URL: https://arxiv.org/abs/2004.09813.

[8] Natalia Segal, Hélène Bonneau-Maynard, and François Yvon. "Traduire la parole: le cas des TED Talks". In: *Revue TAL* 55 (2015), pp. 13–45.

[9] Rico Sennrich, Barry Haddow, and Alexandra Birch. "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1715–1725. DOI: 10.18653/v1/P16-1162. URL: https://aclanthology.org/P16-1162.

[10] Rico Sennrich et al. "Nematus: a Toolkit for Neural Machine Translation". In: *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 65–68. URL: https://aclanthology.org/E17-3017.