Université de Montréal

Utilisation du plongement du domaine pour l'adaptation non supervisée en traduction automatique

par

Xavier Frenette

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

Novembre 2021

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Utilisation du plongement du domaine pour l'adaptation non supervisée en traduction automatique

présenté par

Xavier Frenette

a été évalué par un jury composé des personnes suivantes :

Miklós Csűrös

(président-rapporteur)

Philippe Langlais

(directeur de recherche)

Guy Lapalme

(membre du jury)

Résumé

L'industrie de la traduction utilise de plus en plus des modèles de traduction automatique. Des modèles dits « universels » sont capables d'obtenir de bonnes performances lorsqu'évalués sur un large ensemble de domaines, mais leurs performances sont souvent limitées lorsqu'ils sont testés sur des domaines précis. Or, les traductions doivent être adaptées au style, au sujet et au vocabulaire des différents domaines, en particulier ceux des nouveaux (pensons aux textes reliés à la COVID-19). Entrainer un nouveau modèle pour chaque domaine demande du temps, des outils technologiques spécialisés et de grands ensembles de données. De telles ressources ne sont généralement pas disponibles. Nous proposons, dans ce mémoire, d'évaluer une nouvelle technique de transfert d'apprentissage pour l'adaptation à un domaine précis. La technique peut s'adapter rapidement à tout nouveau domaine, sans entrainement supplémentaire et de façon non supervisée. À partir d'un échantillon de phrases du nouveau domaine, le modèle lui calcule une représentation vectorielle qu'il utilise ensuite pour guider ses traductions. Pour calculer ce plongement de domaine, nous testons cinq différentes techniques. Nos expériences démontrent qu'un modèle qui utilise un tel plongement réussit à extraire l'information qui s'y trouve pour guider ses traductions. Nous obtenons des résultats globalement supérieurs à un modèle de traduction qui aurait été entrainé sur les mêmes données, mais sans utiliser le plongement. Notre modèle est plus avantageux que d'autres techniques d'adaptation de domaine puisqu'il est non supervisé, qu'il ne requiert aucun entrainement supplémentaire pour s'adapter et qu'il s'adapte très rapidement (en quelques secondes) uniquement à partir d'un petit ensemble de phrases.

Mots-clés : traduction automatique, adaptation de domaine, transfert d'apprentissage, apprentissage automatique, apprentissage profond, réseaux de neurones, traitement automatique du langage naturel.

Abstract

Machine translation models usage is increasing in the translation industry. What we could call "universal" models attain good performances when evaluated over a wide set of domains, but their performance is often limited when tested on specific domains. Translations must be adapted to the style, subjects and vocabulary of different domains, especially new ones (the COVID-19 texts, for example). Training a new model on each domain requires time, specialized technological tools and large data sets. Such resources are generally not available. In this master's thesis, we propose to evaluate a novel learning transfer technique for domain adaptation. The technique can adapt quickly to any new domain, without additional training, and in an unsupervised manner. Given a sample of sentences from the new domain, the model computes a vector representation for the domain that is then used to guide its translations. To compute this domain embedding, we test five different techniques. Our experiments show that a model that uses this embedding obtains globally superior performances than a translation model that would have been trained on the same data, but without the embedding. Our model is more advantageous than other domain adaptation techniques since it is unsupervised, requires no additional training to adapt, and adapts very quickly (within seconds) from a small set of sentences only.

Keywords: machine translation, domain adaptation, transfer learning, machine learning, deep learning, neural networks, natural language processing.

Table des matières

Liste des tableaux

Table des figures

Liste des sigles et des abréviations

BTC Bureau de la traduction du Canada

RNN Réseau de neurones récurrent, de l'anglais « Recurrent Neural Network »

TA Traduction automatique

TALN Traitement automatique de la langue naturelle

TAN Traduction automatique neuronale

TAS Traduction automatique statistique

Remerciements

Je remercie premièrement mon directeur de recherche, Philippe Langlais. Il a su trouver le bon équilibre entre me guider et me donner la latitude dont j'avais besoin pour me sentir accompagné, mais également libre d'explorer et d'apprendre. Je remercie également mes collègues du RALI qui étaient toujours disponibles pour m'aider et me guider.

Un merci tout particulier à Brigitte (« *LPMDM* ») qui a ajouté une couche de bonheur et d'amour tout au long de ma recherche, qui m'a épaulé dans les moments de découragement, et qui s'est toujours montrée intéressée et curieuse de mon projet.

Introduction

Motivation

L'utilisation de systèmes de traduction automatique (TA) est en hausse dans l'industrie de la traduction professionnelle (CSA Research, 2021). Elle fut historiquement peu utilisée par les professionnels qui jugeaient défavorablement la qualité des phrases produites (Läubli et Orrego-Carmona, 2017). Mais l'amélioration de la technologie, particulièrement dans la dernière décennie, en fait d'elle maintenant un outil de plus en plus utilisé, aux côtés d'autres instruments comme les mémoires de traduction (ELIA *et al.*, 2020).

La demande pour les services de traduction augmente également, les prévisions indiquant une croissance importante au cours des prochaines années (CSA Research, 2021). Les besoins sont considérables dans une grande variété d'industries disparates, telle celle des réseaux sociaux, du milieu juridique, des jeux vidéos ou de la santé. Un fournisseur de service de traduction doit pouvoir s'adapter à ces différents domaines, à leurs vocabulaires variés et aux différents styles de rédaction. Il doit également pouvoir rapidement offrir ses services dans de nouveaux domaines, comme le démontre l'augmentation importante des besoins en traduction dans le milieu de la santé avec la pandémie de COVID-19 (European Commission, 2021). Ces mêmes qualités de polyvalence et d'adaptation rapide sont aujourd'hui attendues des systèmes de TA.

Vu la variété des domaines, aucun modèle de traduction que l'on pourrait qualifier « d'universel » ne pourra être efficace dans toutes les tâches de traduction. De tels systèmes, comme celui de Google Traduction [1], fonctionnent bien en général, mais ont des difficultés lorsqu'ils sont utilisés sur certains domaines pris individuellement (Caswell et Liang, 2020). Un modèle qui serait spécialisé dans le domaine d'intérêt devient donc plus intéressant qu'un modèle plus universel.

Les meilleurs systèmes de TA d'aujourd'hui utilisent l'apprentissage automatique et une architecture basée sur les réseaux de neurones profonds. L'apprentissage automatique est une stratégie d'intelligence artificielle où un modèle apprend sa tâche (la traduction, par exemple)

1. `https://translate.google.com`

en analysant une banque d'exemples (des paires de phrases traduites professionnellement, par exemple). L'algorithme apprenant tout par lui-même, il n'y a plus besoin de manuellement spécifier au programme des règles de traduction. Cette « automatisation » de la création d'un modèle est un avantage important lorsque vient le temps de créer des systèmes spécialisés dans une large variété de domaines.

Il y a par contre un obstacle important : ces architectures nécessitent énormément de données d'apprentissage (ou « d'entrainement »), souvent des millions, avant de présenter des performances intéressantes (Koehn et Knowles, 2017). De tels ensembles ne sont généralement pas disponibles pour la majorité des domaines. Comment alors utiliser la puissance des réseaux de neurones, mais sur des domaines « pauvres » ? La recherche scientifique propose des solutions principalement basées sur le concept de transfert d'apprentissage. Cette idée stipule que l'on peut créer des modèles spécialisés dans une tâche en utilisant les connaissances acquises lors de l'apprentissage d'autres tâches. Cette vision est assez intuitive : une pianiste pourra accélérer son apprentissage de la guitare si elle utilise ses connaissances déjà acquises en musique.

La majorité des techniques proposées voit ainsi la problématique : pour un nouveau domaine d'intérêt, utilisons une stratégie de transfert d'apprentissage pour entrainer un nouveau modèle qui sera spécialisé dans ce domaine. Cette façon de procéder a un désavantage important : le processus de création d'un modèle et son entrainement doit être recommencé à chaque nouveau domaine. Entrainer un modèle demande des connaissances techniques avancées et des outils technologiques souvent complexes et couteux. Une vision plus efficace serait d'avoir un modèle *unique* qui, à la lecture des textes à traduire, adapte automatiquement et rapidement sa traduction pour le nouveau domaine. Cette façon de faire est d'ailleurs beaucoup plus représentative du fonctionnement humain.

Contributions

La littérature propose peu de ces modèles capables de s'adapter à un nouveau domaine de façon non supervisée (uniquement à partir des textes à traduire) et sans, ou avec un court processus d'entrainement.

Inspirés de Macé et Servan (2019), nous proposons un modèle basé sur l'architecture du Transformer (Vaswani *et al.*, 2017) qui, une fois entrainé, sera à même d'adapter sa traduction uniquement en analysant les phrases à traduire. Le modèle construit, à partir de ces phrases, une représentation vectorielle du domaine et utilise cette information pour guider ses traductions. Aucun entrainement supplémentaire n'est requis pour adapter le modèle. Un tel modèle peut ensuite être utilisé rapidement, et sans un ajustement long et couteux, sur tout nouveau domaine. Différentes stratégies sont possibles pour construire la représentation du

domaine et nous en expérimentons cinq. Le choix des données utilisées pour l'entrainement peut également influer sur les performances possibles du modèle et nous expérimentons sur un ensemble couvrant 25 domaines.

Les résultats de nos expériences démontrent que la technique permet d'améliorer les performances sur certains domaines. Ils font par contre également ressortir des résultats limités sur d'autres domaines et nous proposons une discussion sur le rôle des domaines utilisés durant l'entrainement dans les performances observées.

Nos contributions dans ce mémoire sont :

— Proposition d'une nouvelle technique de transfert d'apprentissage non supervisée pour l'adaptation de domaine.
— Comparaison de cinq techniques de plongement de domaine dans le contexte de la traduction automatique.

Organisation du mémoire

Au chapitre 1, nous introduisons et expliquons les concepts théoriques importants pour bien comprendre la problématique de l'adaptation de domaine en traduction automatique neuronale, et celui de transfert d'apprentissage.

Le chapitre 2 offre un survol de la littérature en traduction automatique, puis en transfert d'apprentissage appliqué à l'adaptation de domaine. Ce parcours des différentes techniques, séparées en apprentissage supervisé et non supervisé, permet de faire ressortir les principales idées qui sous-tendent notre modèle.

Le modèle que nous proposons est introduit et expliqué au chapitre 3. Nous y présentons son intuition sous-jacente, son architecture et détaillons les cinq techniques de plongement de domaine que nous comparerons.

Le chapitre 4 décrit les données avec lesquelles nous travaillons. Nous expliquons leur origine et le prétraitement que nous leur appliquons. Puisque nous souhaitons entrainer notre modèle sur certains domaines et le tester sur d'autres, nous expliquons également la stratégie employée pour la sélection.

Pour bien mesurer l'amélioration des performances de notre modèle, nous le comparons à des solutions existantes. Le chapitre 5 présente les modèles de comparaison qui ont été choisis et les critères qui ont mené à leur choix.

Nous présentons les résultats de nos expériences au chapitre 6. Après description des métriques de comparaison quantitative utilisées (BERTScore, BLEU, chrF++), nous évaluons les performances de notre modèle et celles des modèles de comparaison. Nous proposons une

analyse détaillée de notre modèle et présentons une discussion sur le choix des domaines utilisés pour l'entrainement.

Chapitre 1

Concepts théoriques

Créer un modèle de traduction automatique neuronale (TAN) spécialisé dans un domaine est une tâche complexe. Ce chapitre explique les concepts théoriques permettant de bien comprendre la problématique et les solutions que nous explorerons dans la suite de notre mémoire. Nous commençons par présenter les concepts de traduction automatique, de TAN, de domaine et de corpus d'entrainement. Nous décrivons ensuite les deux principaux obstacles à la création de modèles de TAN spécialisés dans un domaine : leur gourmandise en données d'entrainement et leur sensibilité aux différences entre les domaines. Finalement, nous présentons une famille de solutions à ces deux problèmes : le transfert d'apprentissage, un concept sur lequel sont basées les techniques d'adaptation de domaine qui seront explorées dans la suite du mémoire.

1.1. La traduction automatique neuronale

Nous présentons premièrement le concept général de traduction automatique et son historique avant de définir ce qu'est la traduction automatique neuronale.

1.1.1. La traduction automatique

La traduction automatique (TA) est un processus informatique où un texte dans une langue **source** est traduit vers une langue **cible**. Elle peut s'appliquer aux langues artificielles (les langages de programmation, par exemple), mais elle est surtout utilisée pour les langues naturelles (de l'anglais vers le français, par exemple). Pour être efficace et utile, un bon système de traduction automatique devrait être complet, sans nécessiter de prétraitement ni de postédition humaine de la traduction. De plus, une bonne traduction doit adéquatement capturer le sens du document original, tout en étant facile à lire. Ces concepts d'adéquation et de fluidité sont reconnus depuis longtemps (voir, par exemple, ALPAC, 1966, appendice

10), et aujourd'hui encore (voir, par exemple, Koehn, 2004), comme les deux principaux objectifs d'un bon modèle de TA.

Dans ce mémoire, nous nous intéressons uniquement aux modèles de TA bilingue, qui traduisent toujours de la même langue source vers la même langue cible. Il est à noter qu'il existe cependant des modèles de TA multilingue, qui peuvent traduire entre plusieurs paires de langues (voir, par exemple, Johnson *et al.*, 2017; Lample et Conneau, 2019), et que ce domaine de recherche est en expansion.

L'intérêt dans la TA est pratiquement né avec l'informatique. Ayant pris plusieurs formes au cours des six dernières décennies, la recherche a connu des hauts et des bas depuis les premiers dictionnaires « mécaniques » jusqu'à la traduction « statistique » avant de mener à la traduction « neuronale » qui sera explorée dans la prochaine section.

Dès l'arrivée des premiers ordinateurs programmables dans les années 40, un intérêt pour la « traduction mécanique » se développe. Warren Weaver, bien connu pour ses travaux en théorie de la communication avec Claude Shannon, et Andrew Booth, un cristallographe anglais, explorent les possibilités de la TA à la fin des années 1940 (Weaver et others, 1955). Les années 50 et 60 voient le développement de la TA dite « par règles ». Inspirés par le développement parallèle de la théorie linguistique, les informaticiens développent des systèmes qui utilisent règles linguistiques et dictionnaires électroniques pour générer des traductions. Les systèmes sont surtout utilisés pour la traduction entre l'anglais et le russe, guerre froide oblige, et pour les domaines scientifiques et techniques, là où les différences culturelles et les contextes variables sont moins prononcés (Hutchins, 2007). Mais malgré l'intérêt de la communauté scientifique, la recherche subira un contrecoup avec la publication du rapport de l'ALPAC.

En 1966, le Automatic Language Processing Advisory Committee (ALPAC) produit un rapport (ALPAC, 1966) qui a des effets dramatiques sur la recherche en TA. Commandé par le gouvernement américain pour évaluer l'état de la TA, le rapport se dit déçu du peu d'avancement dans la recherche et de la mauvaise qualité des traductions des modèles testés. Il constate que malgré des années de recherche, il demeure plus onéreux de faire de la traduction par une machine que par un humain. De plus, le rapport est sceptique quant aux possibilités futures du domaine. Bien que le comité suggère de continuer la recherche pour son intérêt scientifique, le gouvernement américain réduit drastiquement son financement pour la prochaine décennie et la communauté reste avec une impression d'échec (Hutchins, 2003).

Malgré cette perte d'intérêt général, les années suivantes verront quelques réussites intéressantes, comme le système TAUM-Météo développé dans les années 70 par le projet Traduction automatique de l'Université de Montréal (TAUM). Ce système, considéré comme une

percée en TA (Isabelle et Bourbeau, 1985; Hutchins, 2007), mènera au développement d'un modèle de traduction de prévision météo qui sera utilisé par Environnement Canada jusqu'en 2001. Mais la principale percée survient avec le développement de la traduction automatique statistique (TAS).

Avec la TAS, la tâche de traduire un texte n'est plus vue comme une série de règles à appliquer, mais comme un problème probabiliste qui se résout par la statistique. Cette vision, basée sur des idées de la théorie de l'information et de la cryptographie, n'est pas nouvelle. Elle est proposée dès 1949 par Warren Weaver : « When I look at an article in Russian, I say : "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode ." » (Weaver et others, 1955). Mais les besoins intenses en calculs et en données de cette technique ont poussé les chercheurs vers d'autres avenues pendant plusieurs décennies. C'est à la fin des années 1980 que la TAS revient à l'avant-scène. L'utilisation fructueuse des méthodes statistiques en reconnaissance vocale et en traitement de la langue naturelle inspire une équipe d'IBM à les appliquer à la TA. Les modèles de TAS qu'ils développent présentent des résultats très intéressants (Brown *et al.*, 1988, 1990) qui vont démarrer une nouvelle ère dans le domaine.

En TAS, la traduction est vue comme un problème de décryptage. En cryptographie, un message « compréhensible » $M \in \mathcal{M}$ est encrypté en une version « incompréhensible » Ξ. L'objectif est de retrouver le message original. D'un point de vue probabiliste, tout message $m \in \mathcal{M}$ est un décryptage possible du message original, mais certains sont plus probables que d'autres. La tâche consiste donc à trouver le message $\hat{M}$ le plus probable.

$$\hat{M} = \underset{m \in \mathcal{M}}{\operatorname{argmax}}\, p(m|\Xi) \tag{1.1.1}$$

Avec un peu de gymnastique mentale, nous pouvons interpréter la traduction automatique comme un problème cryptographique. Pour une tâche de traduction de l'anglais vers le français, nous pouvons imaginer un document français $f \in \mathcal{F}$ qui aurait été « encrypté » vers une version anglaise a que l'on reçoit. Notre tâche consiste donc à trouver le document français $\hat{f} \in \mathcal{F}$ qui serait la meilleure (d'un point de vue probabiliste) version « décryptée » de la version anglaise.

$$\hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmax}}\, p(f|a) \tag{1.1.2}$$

Par le théorème de Bayes, l'équation 1.1.2 peut se réécrire :

$$\hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmax}}\, p(a|f)p(f) \tag{1.1.3}$$

L'intérêt de cette réécriture est qu'elle divise le problème en deux sous-problèmes qui peuvent être résolus séparément. Si nous avons accès à un corpus d'exemples de traductions (des textes anglais et leur traduction française associée) suffisamment grand, nous pouvons calculer une approximation du premier terme de l'équation 1.1.3. Le deuxième terme peut être approximé avec un corpus de textes unilingues français.

Les modèles de TAS cherchent donc à résoudre l'équation 1.1.2 en modélisant les termes de l'équation 1.1.3 et en recherchant la traduction la plus probable. Contrairement aux modèles antérieurs, il n'est plus nécessaire de spécifier un ensemble de règles de traductions, le modèle apprenant par lui-même sa paramétrisation. Dans les faits, un modèle de TAS sera généralement une combinaison de plusieurs modules, chacun optimisé séparément (Bahdanau *et al.*, 2014).

Les modèles de TAS resteront les modèles les plus efficaces en TA pendant plus de vingt ans, jusqu'au développement de la traduction automatique par réseaux de neurones.

1.1.2. Traduction automatique par réseaux de neurones

La traduction automatique à l'aide de réseaux de neurones profonds, plus communément appelée traduction automatique neuronale (TAN), est proposée à partir de 2013 (Kalchbrenner et Blunsom, 2013; Cho *et al.*, 2014a; Sutskever *et al.*, 2014). Les réseaux de neurones profonds sont des modèles d'apprentissage automatique qui atteignent déjà à cette époque d'excellentes performances dans divers problèmes, comme la reconnaissance vocale ou la reconnaissance visuelle (Sutskever *et al.*, 2014). Leur capacité à découvrir des structures complexes dans des données à haute dimension (LeCun *et al.*, 2015) et leur habilité à pouvoir faire un nombre arbitraire de calculs parallèles en peu d'étapes (Sutskever *et al.*, 2014) leur permettent de résoudre mieux qu'avant, et plus rapidement, des problèmes complexes comme la traduction automatique.

Tout comme les modèles de TAS, un modèle de TAN résout une tâche de traduction automatique en cherchant une solution à l'équation probabiliste 1.1.2. Et comme dans la TAS, le modèle approxime la distribution $p(f|a)$ à partir d'un corpus de textes parallèles.

Les modèles neuronaux ont plusieurs avantages par rapport aux modèles statistiques. Tout d'abord, il s'agit d'une approche que l'on pourrait qualifier de « complète » : le système peut être entrainé de bout en bout sans que l'on ait à créer des systèmes d'alignements, des règles de traductions ou des algorithmes de décodage complexes qui sont souvent requis en TAS (Cho *et al.*, 2014a; Chu *et al.*, 2018). Ils ne nécessitent qu'une fraction de la mémoire généralement requise par les modèles de TAS (Cho *et al.*, 2014a). Par l'utilisation de réseaux de neurones récurrents, et plus tard le Transformer (Vaswani *et al.*, 2017), un modèle de

TAN peut facilement capturer les relations « long terme » entre les mots éloignés, ce qui est plus difficile en TAS (Tu *et al.*, 2016).

Rapidement, les modèles de TAN surpassent les modèles de TAS dans certaines tâches de traduction automatique (Bojar *et al.*, 2016) et ils sont aujourd'hui la technique de choix pour la création d'un système de traduction.

1.2. Le domaine et la tâche

Malgré les avantages et les performances impressionnantes des modèles neuronaux, créer un système spécialisé dans un domaine demeure une tâche complexe. Avant de voir pourquoi, nous expliquons ce qu'est un domaine et, dans la prochaine section, la forme sous laquelle il se présente au modèle : le corpus d'entrainement.

Le concept de domaine n'est pas limité à la traduction automatique. Il peut se retrouver dans toute tâche d'apprentissage automatique. Par exemple, pour un système de reconnaissance d'images de chien, l'espace des photos et celui des dessins de chiens pourraient être considérés comme deux domaines. En reconnaissance de la voix, l'espace des extraits lus par un accent précis peut être considéré comme un domaine. En traduction automatique, un modèle de traduction d'avis pourrait considérer les avis de films et ceux de livres comme deux domaines.

Deux caractéristiques importantes ressortent de ces exemples. Premièrement, un domaine est défini selon l'objectif de la tâche à accomplir. Deux tâches différentes auront des règles différentes quant à la répartition des données en domaines. Deuxièmement, nous remarquons qu'un domaine partage souvent des caractéristiques avec d'autres domaines. C'est ce partage d'informations qui mènera à la famille de solutions que nous verrons à la section §1.5.

Malgré la généralité du concept, nous pouvons quand même lui donner une définition formelle. Nous examinerons ensuite plus spécifiquement l'idée de domaine en traduction automatique.

1.2.1. Définition formelle

Différentes définitions formelles de domaine existent (voir par exemple Ben-David *et al.*, 2010; Weiss *et al.*, 2016; Kouw et Loog, 2018). Nous utiliserons une définition qui distingue le domaine de la tâche (Pan et Yang, 2010; Weiss *et al.*, 2016; Zhuang *et al.*, 2021).

Un domaine $\mathcal{D}$ est un couple $\{\mathcal{X},P_{\mathcal{X}}\}$ (Pan et Yang, 2010; Zhuang *et al.*, 2021) où $\mathcal{X}$ est un espace source (d'entrée) et $P_{\mathcal{X}}$ la distribution de probabilité marginale de cet espace. En traduction automatique, $\mathcal{X}$ pourrait, par exemple, être l'espace de tous les documents en anglais, et $P_{\mathcal{X}}$ serait la probabilité de distribution pour chaque document, en d'autres

mots la probabilité d'observer ce document si on sélectionnait au hasard un document du domaine.

Pour un domaine donné, une tâche $\mathcal{T}$ est un couple $\{\mathcal{Y},f\}$ (Pan et Yang, 2010; Zhuang *et al.*, 2021) où $\mathcal{Y}$ est un espace cible (de sortie) et $f : \mathcal{X} \mapsto \mathcal{Y}$ est une fonction prédictive. C'est cette fonction que le modèle cherche à apprendre. En traduction automatique, $\mathcal{Y}$ pourrait, par exemple, être l'espace de tous les documents en français. La fonction de décision f est, quant à elle, soit une distribution de probabilité conditionnelle $P_{\mathcal{Y}|\mathcal{X}}$ qui retourne la probabilité qu'un document cible soit la traduction d'un document source, soit le `argmax` de cette fonction, comme à l'équation 1.1.2.

Pour bien comprendre chacune des composantes définies ci-dessus, donnons l'exemple de deux tâches de traduction de l'anglais vers le français, une pour le domaine « médical », une autre pour le domaine « photographie ». L'espace source est le même pour les deux domaines : celui de tous les documents anglais. Mais leur distribution de probabilité n'est pas égale. En effet, il est beaucoup plus probable d'observer la phrase « *Polyps of the nose* » dans le domaine « médical » que « photographie ». L'espace cible est également le même pour les deux tâches : l'espace de tous les documents français. Mais la fonction prédictive est différente. En effet, étant donné la phrase source « *An important shot* », la phrase cible « *Un vaccin important* » est plus probable pour la tâche « médical » que la phrase « *Une photo importante* », alors que c'est l'inverse pour la tâche « photographie ». Mathématiquement, nous avons les équations suivantes pour notre exemple (où (m) est pour « médical » et (p) est pour « photographie ») :

$$\mathcal{X}^{(m)} = \mathcal{X}^{(p)} \tag{1.2.1}$$

$$P_{\mathcal{X}}^{(m)} \neq P_{\mathcal{X}}^{(p)} \tag{1.2.2}$$

$$\mathcal{Y}^{(m)} = \mathcal{Y}^{(p)} \tag{1.2.3}$$

$$P_{\mathcal{Y}|\mathcal{X}}^{(m)} \neq P_{\mathcal{Y}|\mathcal{X}}^{(p)} \tag{1.2.4}$$

En particulier :

$$P_{\mathcal{X}}^{(m)}(\text{"Polyps of the nose"}) > P_{\mathcal{X}}^{(p)}(\text{"Polyps of the nose"}) \tag{1.2.5}$$

$$P_{\mathcal{Y}|\mathcal{X}}^{(m)}(\text{"Un vaccin important"}|\text{"An important shot"}) > P_{\mathcal{Y}|\mathcal{X}}^{(m)}(\text{"Une photo importante"}|\text{"An important shot"}) \tag{1.2.6}$$

$$P_{\mathcal{Y}|\mathcal{X}}^{(p)}(\text{"Un vaccin important"}|\text{"An important shot"}) < P_{\mathcal{Y}|\mathcal{X}}^{(p)}(\text{"Une photo importante"}|\text{"An important shot"}) \tag{1.2.7}$$

Deux domaines sont considérés comme différents si leur espace source ou leur distribution marginale est différent, et deux tâches sont différentes si leur espace cible ou leur fonction objective est différent.

La littérature en traduction automatique ne différencie souvent pas explicitement le domaine de la tâche, celle-ci étant souvent sous-entendue. Afin d'alléger le texte et de rester proches de la nomenclature utilisée dans le domaine, nous laisserons également implicite la différence, sauf exception. Ainsi, « un modèle spécialisé pour le domaine A » suppose que la distribution de l'espace d'entrée ($P_{\mathcal{X}}$) est celle du domaine A et que la tâche consiste à générer des traductions qui correspondent au domaine A dans la langue cible.

Notons également que, dans notre mémoire, nous nous intéressons aux modèles qui traduisent toujours de la même langue source vers la même langue cible. Il sera donc toujours sous-entendu que l'espace d'entrée $\mathcal{X}$ sera le même pour tous les domaines et que l'espace de sortie $\mathcal{Y}$ sera également identique pour toutes les tâches.

1.2.2. Définition qualitative du domaine en traduction automatique

Nous avons souligné en introduction de ce chapitre que le domaine dépend de la tâche. En traduction automatique, comment reconnaitre un domaine ? Nous avons défini mathématiquement que deux domaines sont différents si leurs distributions de probabilité ne sont pas égales. Mais en réalité, nous n'avons pas accès à cette distribution. Il importe donc de pouvoir distinguer qualitativement si deux corpus font partie du même domaine ou non.

Différentes définitions de ce qu'est un domaine dans le contexte de la traduction automatique ont été proposées. Dans une définition qu'ils qualifient de « commune », Koehn et Knowles (2017) le décrivent comme « un corpus tiré d'une source spécifique, et qui peut varier d'un autre domaine en sujet, genre, style, niveau de formalité, etc. ». van der Wees *et al.* (2015) limite la qualification à deux caractéristiques importantes, en plus de sa source : son sujet et son genre. Le sujet (par exemple : nouvelles, textes médicaux, textes juridiques) peut être général ou précis et se révèle souvent par la distribution du vocabulaire (Saunders, 2021). Le genre consiste en la fonction du texte, sa syntaxe et son style. Ce dernier concept est orthogonal au sujet : deux documents traitant du même sujet peuvent avoir deux genres différents. Un même document peut contenir plusieurs sujets et genres, donc contenir plusieurs domaines. Vice-versa, des textes d'un même domaine peuvent se retrouver dans divers documents, de provenances variées (Saunders, 2021). Aharoni et Goldberg (2020) quant à eux appellent à définir le domaine non pas selon sa source, mais au niveau de la phrase. Différents documents peuvent contenir différentes phrases avec des traits similaires. Ils proposent comme domaine un regroupement de phrases « similaires ». Ils ne précisent par contre pas la notion de « similarité ».

1.3. Le corpus d'entrainement

Tout le processus de traduction se situe au niveau de la fonction objective de la tâche, soit la densité conditionnelle $P_{\mathcal{Y}|\mathcal{X}}$. Pour tout document de l'espace source du domaine, cette densité retourne la probabilité qu'un document de l'espace cible soit une traduction. La « meilleure » traduction étant celle avec la plus haute probabilité.

Cette distribution est inconnue. Nous utilisons un modèle de TAN comme approximation de cette distribution et le rôle de l'algorithme d'apprentissage est de trouver les paramètres du modèle lui permettant de s'approcher le plus possible de la distribution réelle. L'apprentissage se fait au moyen d'un corpus d'entrainement qui est analysé par l'algorithme pour rechercher les meilleurs paramètres du modèle.

Un corpus d'entrainement E est un ensemble de paires de documents sources et leur traduction associée : $E = \{(x_i,y_i) : i = 1,...,n\}$. On parle d'un corpus « parallèle ». Cet ensemble n'est ni le domaine ni la tâche. Il s'agit d'un échantillon tiré de la distribution jointe $P_{\mathcal{X},\mathcal{Y}}$. En effet, les documents sources x_i sont tirés selon la distribution de probabilité du domaine $P_{\mathcal{X}}$, et la traduction associée est tirée de la distribution conditionnelle $P_{\mathcal{Y}|\mathcal{X}}$. Donc chaque paire (x_i,y_i) est tirée de la distribution $P_{\mathcal{X}}P_{\mathcal{Y}|\mathcal{X}} = P_{\mathcal{X},\mathcal{Y}}$.

Créer un modèle de TAN spécialisé dans un domaine consiste donc, en théorie, à fournir à l'algorithme d'apprentissage un corpus d'entrainement pour ce domaine. Malheureusement, en réalité, cela s'avère généralement être une tâche difficile.

1.4. Les difficultés de la TAN avec les petits domaines

Nous présentons les deux principaux obstacles à la création de modèles de TAN performants et spécialisés pour un domaine particulier : la nécessité d'une quantité de données d'entrainement souvent plus importante que ce qui est disponible, et l'inefficacité générale d'un modèle entrainé pour un domaine, mais utilisé pour un autre.

1.4.1. Avidité pour les données

Les réseaux de neurones profonds, la structure sous-jacente des modèles de TAN, ont la capacité d'apprendre des relations très complexes entre une entrée et une sortie. Ils peuvent, par exemple, découvrir la relation qui existe entre une phrase en anglais et sa traduction française. Le réseau apprend ces relations à l'aide du corpus d'entrainement. Celui-ci, s'il est suffisamment grand, va généralement posséder une représentation complète de ces relations. Mais s'il est trop petit par rapport à la capacité d'apprentissage du modèle, celui-ci peut tomber en surapprentissage.

Le surapprentissage (ou « *overfitting* » en anglais) est un état où le modèle s'est surspécialisé sur l'ensemble d'entrainement, au détriment de ses performances sur la distribution réelle des données. Plutôt que d'avoir découvert les relations générales présentent dans la distribution, le modèle a appris « par cœur » les relations spécifiques à l'ensemble d'entrainement. Bien que le modèle soit « performant » sur les exemples vus à l'entrainement, ses performances sont généralement faibles sur tout autre corpus tiré du domaine.

Les modèles de TAN, étant basés sur les réseaux de neurones profonds, sont très sensibles au surapprentissage et nécessitent de grands ensembles d'entrainement. Les résultats de Bojar *et al.* (2016) annonçaient les modèles de TAN comme supérieurs aux modèles de TAS, mais seulement lorsque de grands corpus (par exemple, des millions de phrases) étaient utilisés. Zoph *et al.* (2016) ont démontré qu'un modèle de traduction statistique standard obtient de meilleures performances qu'un modèle de TAN lorsque l'ensemble d'entrainement est limité.

La principale solution au surapprentissage est d'utiliser un ensemble d'entrainement suffisamment grand. Koehn et Knowles (2017) démontrent qu'une fois atteint un certain seuil, un modèle de TAN réussit à dépasser les performances d'un TAS. On parle généralement de quelques millions de paires de phrases parallèles (des phrases dans la langue source, chacune associée à sa traduction humaine). Un tel nombre est nécessaire parce qu'un ensemble de données contient toujours un peu de bruit : le même mot peut être traduit différemment, avec un genre ou un nombre différent, utilisé dans différents contextes, etc. L'ensemble doit donc est assez grand afin de permettre au modèle de distinguer le bruit des relations sous-jacentes (Srivastava *et al.*, 2014).

Les modèles de TAN ont la capacité d'offrir les meilleures performances, il faut seulement pouvoir les nourrir !

Trouver des ensembles de millions de phrases parallèles n'est par contre pas une tâche aisée. Sauf exception, il est rare de trouver plus que quelques centaines ou milliers de phrases parallèles pour un domaine. Pour la majorité des domaines, il n'existe tout simplement aucun corpus parallèle. Les difficultés sont encore plus grandes pour les langues avec peu de locuteurs ou pour les paires de langues rarement traduites ensemble.

Créer de nouveaux corpus parallèles est une tâche très onéreuse en temps et en argent, il ne s'agit généralement pas d'une solution réaliste.

La réalité est, en conséquence, que pour la majorité des domaines, les corpus d'entrainement parallèles sont soit trop petits par rapport à la capacité du modèle, soit inexistants. Il est donc rare de pouvoir entrainer un modèle compétent uniquement sur un ensemble de données du domaine d'intérêt.

S'il n'est pas possible de trouver, ou de créer, un ensemble de données suffisamment grand pour un domaine spécifique, peut-être pourrait-on simplement utiliser les données d'un autre

domaine, ou une combinaison d'autres domaines, pour l'entrainement ? Comme nous allons le voir, ceci ne permet pas non plus de créer des modèles de qualité.

1.4.2. Sensibilité aux différences entre les domaines

Un modèle de TAN entrainé sur un domaine précis n'est pas universel : ses performances seront généralement faibles lorsqu'utilisé sur un domaine différent (Koehn et Knowles, 2017). Il s'agit d'une concrétisation d'un problème commun en apprentissage automatique : l'algorithme d'apprentissage est construit selon l'hypothèse que les données d'entrainement et les données de test proviennent de la même distribution. Une différence entre ces distributions va généralement entrainer une perte lorsque le modèle sera utilisé avec les données de test. Nous démontrons ici mathématiquement pour la traduction automatique que si deux domaines sont différents, alors leurs données proviendront de deux distributions différentes.

À partir des définitions de la section 1.2, soit deux domaines A et B, le premier utilisé pour l'entrainement, et le second pour le test. Soit également leur tâche respective. Chaque domaine est une paire $\{\mathcal{X},P_{\mathcal{X}}\}$, et chaque tâche est une paire $\{\mathcal{Y},P_{\mathcal{Y}|\mathcal{X}}\}$. Avec la même supposition que $\mathcal{X}^{(A)} = \mathcal{X}^{(B)}$ et que $\mathcal{Y}^{(A)} = \mathcal{Y}^{(B)}$, alors les deux domaines sont différents si et seulement si $P_{\mathcal{X}}^{(A)} \neq P_{\mathcal{X}}^{(B)}$ ou $P_{\mathcal{Y}|\mathcal{X}}^{(A)} \neq P_{\mathcal{Y}|\mathcal{X}}^{(B)}$. Par le théorème de Bayes, nous avons que :

$$P_{\mathcal{X},\mathcal{Y}}^{(A)} = P_{\mathcal{X}}^{(A)} P_{\mathcal{Y}|\mathcal{X}}^{(A)} \tag{1.4.1}$$

$$P_{\mathcal{X},\mathcal{Y}}^{(B)} = P_{\mathcal{X}}^{(B)} P_{\mathcal{Y}|\mathcal{X}}^{(B)} \tag{1.4.2}$$

et donc que, en général, deux domaines seront différents si $P_{\mathcal{X},\mathcal{Y}}^{(A)} \neq P_{\mathcal{X},\mathcal{Y}}^{(B)}$.

Nous avons vu à la section §1.3 que le corpus d'entrainement est un échantillon tiré de la distribution $P_{\mathcal{X},\mathcal{Y}}^{(A)}$. Par le même raisonnement, les échantillons pour le test seront tirés de la distribution $P_{\mathcal{X},\mathcal{Y}}^{(B)}$.

Nous voyons donc que la distribution utilisée pour l'entrainement est différente de celle du test. Le modèle entrainé sur le domaine A sera biaisé s'il est utilisé sur le domaine B. Ce phénomène de différence entre les deux distributions est appelé *data shift* (un déplacement des données) et, en traduction automatique, il se présente généralement sous deux formes : le *concept shift* et le *covariate shift*.

Le *concept shift* (Kouw et Loog, 2018) est lorsque $P_{\mathcal{Y}|\mathcal{X}}^{(A)} \neq P_{\mathcal{Y}|\mathcal{X}}^{(B)}$. Pour un document de test du domaine B, tiré de la distribution $P_{\mathcal{X}}^{(B)}$, un modèle entrainé sur le domaine A produira la traduction la plus probable selon la distribution $P_{\mathcal{Y}|\mathcal{X}}^{(A)}$, et non selon la distribution $P_{\mathcal{Y}|\mathcal{X}}^{(B)}$. Le *covariate shift* (Kouw et Loog, 2018; Shimodaira, 2000) est lorsque $P_{\mathcal{X}}^{(A)} \neq P_{\mathcal{X}}^{(B)}$. Dans ce cas, le document de test tiré depuis $P_{\mathcal{X}}^{(B)}$ pourrait se retrouver dans une zone de faible

densité dans la distribution $P_{\mathcal{X}}^{(A)}$. Selon le modèle entrainé sur A, il s'agirait alors d'un exemple peu probable. Les traductions générées pour les exemples de faible densité sont instables, car l'algorithme d'apprentissage se concentre plus sur les zones de hautes densités (car plus représentées dans l'ensemble d'entrainement). L'instabilité cause généralement des traductions de mauvaise qualité.

Dans la réalité, la différence entre deux domaines est généralement une combinaison de *concept shift* et de *covariate shift*. Utiliser un modèle sur un domaine alors qu'il a été entrainé sur un autre produira généralement de mauvais résultats. Il ne s'agit donc pas d'une solution possible lorsque le domaine d'intérêt ne possède pas suffisamment de données d'entrainement.

1.5. Transfert d'apprentissage et adaptation de domaine

Deux principales raisons complexifient la création d'un modèle de TAN spécialisé dans un domaine : la quantité de données n'est souvent pas suffisante pour entrainer un modèle sur ce domaine, et utiliser un modèle entrainé sur un autre domaine ne donne généralement pas de bons résultats. Une solution existe : le transfert d'apprentissage.

Le transfert d'apprentissage est toute technique qui améliore l'apprentissage d'une tâche cible en utilisant les connaissances acquises dans l'apprentissage d'un domaine source (Torrey et Shavlik, 2009; Pan et Yang, 2010; Weiss *et al.*, 2016). Il s'agit donc de l'apprentissage d'une tâche cible, mais avec une source d'information supplémentaire : la connaissance d'un domaine ou d'une tâche reliée.

Le transfert d'apprentissage est inspiré de la capacité humaine à utiliser nos connaissances acquises pour résoudre de nouvelles tâches plus rapidement ou plus efficacement. Par exemple, une pianiste professionnelle pourra apprendre à jouer de l'accordéon plus rapidement qu'une novice en musique. De même, un locuteur francophone apprendra une autre langue latine plus facilement qu'un locuteur chinois. Dans ces situations, la personne utilise des connaissances communes aux deux tâches pour accélérer l'apprentissage.

Ce concept de tâche commune se retrouve aussi dans la traduction automatique. Deux tâches de traduction de l'anglais vers le français, une pour un domaine A et l'autre pour un domaine B, partagent une tâche commune : celle de la traduction de l'anglais vers le français. Ainsi, un modèle qui cherche à apprendre le domaine B pourrait utiliser les connaissances acquises par un modèle sur le domaine A pour améliorer son apprentissage.

Le transfert d'apprentissage permet d'améliorer l'apprentissage de la tâche cible de trois principales façons (voir la figure 1.1) : démarrer l'apprentissage dans la tâche cible avec une performance initiale supérieure à un modèle ignorant, favoriser une courbe d'apprentissage

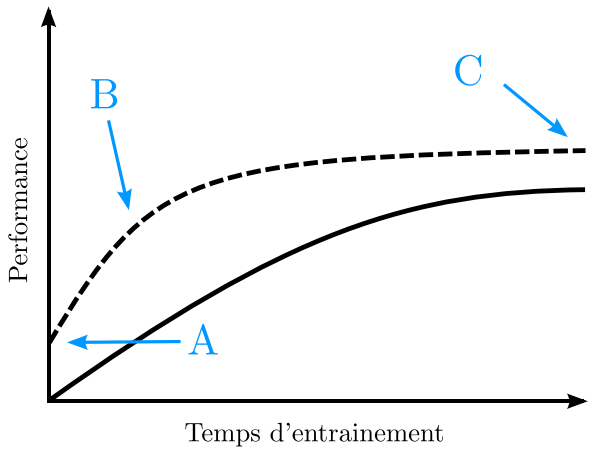

Figure 1.1. L'utilisation du transfert d'apprentissage (ligne pointillée) peut améliorer de trois façons l'apprentissage par rapport à l'apprentissage sans transfert (ligne pleine). (A) Performance initiale supérieure. (B) Apprentissage plus rapide. (C) Performance finale supérieure. Image inspirée de Torrey et Shavlik (2009).

plus rapide, et obtenir une performance finale plus élevée par rapport à la performance finale d'un modèle sans transfert (Torrey et Shavlik, 2009).

Les modèles basés sur des réseaux de neurones profonds, comme les modèles de TAN, se prêtent particulièrement bien au transfert d'apprentissage (Bengio, 2012a). Leur structure en multiples couches leur permet d'apprendre différents niveaux de représentations abstraites de la donnée d'entrée. Les représentations bas niveau des premières couches sont utilisées et combinées par les couches supérieures pour créer des représentations de haut niveau.

Certaines des représentations apprises peuvent être communes à d'autres tâches. Transférer ces connaissances à d'autres tâches peut faciliter l'apprentissage dans des situations où le corpus d'entrainement est trop petit pour apprendre tous les niveaux de représentation (Bengio, 2012a).

Les techniques que nous explorons dans le chapitre 2 et celle que nous utilisons à partir du chapitre 3 sont toutes des techniques de transfert d'apprentissage.

1.5.1. Adaptation de domaine

La littérature sur le transfert d'apprentissage utilise parfois le terme « adaptation de domaine ». Ce même terme est aussi souvent utilisé en traduction automatique. Sa définition n'est pas claire et varie selon les auteurs.

Weiss *et al.* (2016) limitent le concept d'adaptation de domaine à altérer le domaine source pour rapprocher sa distribution de celle du domaine cible. Arnold *et al.* (2007) considèrent que l'adaptation de domaine est lorsque la tâche reste la même, mais que les domaines

source et cibles changent. Kouw et Loog (2018) limitent l'adaptation de domaine au cas où les espaces source et cible demeurent les mêmes, mais les probabilités de distribution changent : soit une des densités marginale $P_{\mathcal{X}}$ et $P_{\mathcal{Y}}$, soit la probabilité jointe $P_{\mathcal{X},\mathcal{Y}}$.

En traduction automatique, Saunders (2021) considère l'adaptation de domaine comme tout processus qui cherche à améliorer la traduction d'un système existant pour un certain genre ou sujet sans réentrainer complètement le modèle. Pour Chu et Wang (2018), il s'agit de l'utilisation d'un corpus de données parallèles extradomaine pour améliorer l'apprentissage lorsque seul un corpus unilingue est disponible dans le domaine d'intérêt. Finalement, le terme est parfois synonyme de « transfert d'apprentissage » (Pan et Yang, 2010).

Pour notre mémoire, nous nous intéressons uniquement aux modèles qui traduisent toujours du même espace source vers le même espace cible. Notre définition d'adaptation de domaine est semblable à celle de Kouw et Loog (2018) :

L'adaptation de domaine pour la traduction automatique, un cas particulier du transfert d'apprentissage, est toute technique permettant d'améliorer l'apprentissage d'une tâche de traduction pour un domaine cible à partir des connaissances implicites d'un ou plusieurs domaines sources ou de l'apprentissage de leur tâche de traduction. Elle s'applique dès que la distribution de l'espace source est différente de celle de l'espace cible ($P_{\mathcal{X}}^{(source)} \neq P_{\mathcal{X}}^{(cible)}$) ou que les distributions conditionnelles sont différentes ($P_{\mathcal{Y}|\mathcal{X}}^{(source)} \neq P_{\mathcal{Y}|\mathcal{X}}^{(cible)}$).

Le prochain chapitre survole les techniques de transfert d'apprentissage existantes en traduction automatique.

Chapitre 2

Revue de littérature

Nous présentons dans ce chapitre la littérature récente sur les concepts clés que nous avons vus au chapitre 1. Ces articles seront également mis dans le contexte du modèle que nous proposons au chapitre 3.

La section §2.1 porte sur l'architecture des modèles de traduction neuronale. La section §2.2 explore les techniques de transfert d'apprentissage appliquées à notre problématique. La section §2.3 survole des méthodes qui utilisent le contexte local et global de la phrase, un concept qui a une place importante dans notre modèle.

Pour simplifier la lecture, nous utiliserons dans ce chapitre, et dans le reste du mémoire, les termes « phrase » et « mot » de façon générique. Une « phrase » est tout texte donné en entrée au modèle de traduction, et est composée d'un ensemble de mots, de ponctuations et de symboles. Il ne s'agit pas nécessairement d'une phrase dans le sens linguistique du terme. Une phrase est divisée en « mots », les unités de la phrase. La division dépend du modèle et une unité pourrait être un mot linguistique, une partie d'un tel mot, un signe de ponctuation, etc. Par exemple, un modèle pourrait choisir de diviser la phrase suivante « *Bonjour, l'ami.* » en six *mots* : « *Bon – jour – , – l' – ami – .* ». Nous utiliserons également l'expression « adaptation de domaine » telle que définie à la section 1.5.1.

2.1. Modèles de traduction neuronale

Les réseaux de neurones simples ne sont pas adaptés à la traduction automatique. La donnée d'entrée se doit d'être d'une taille précise et la sortie du réseau est également d'une taille prédéterminée. Cependant, une phrase à traduire et sa traduction sont chacune une séquence de mots de *taille variable*. Deux phrases peuvent être composées d'un nombre différent de mots, et une traduction n'a généralement pas le même nombre de mots que sa phrase source.

Des architectures particulières de réseaux de neurones, adaptées à ces problèmes dit de « séquence-à-séquence », ont donc été développées.

2.1.1. Réseaux récurrents

Les premiers modèles de TAN résolvent ce problème par une approche *encodeur-décodeur* (Kalchbrenner et Blunsom, 2013; Sutskever *et al.*, 2014; Cho *et al.*, 2014b). En particulier Sutskever *et al.* (2014) et Cho *et al.* (2014b) proposent une architecture basée sur les réseaux de neurones récurrents (RNN, de l'anglais « *recurrent neural network* »). Un premier RNN, appelé « encodeur », lit la phrase d'entrée mot-à-mot pour construire une représentation vectorielle de taille fixe. Cette représentation est ensuite utilisée par un autre RNN, appelé « décodeur », qui génère une hypothèse de traduction mot-à-mot, jusqu'à générer le mot spécial indiquant la fin de la phrase [1]. Ces deux modules sont entrainés conjointement, et de manière supervisée, dans une tâche de traduction qui a pour objectif de maximiser la log-probabilité de l'hypothèse de traduction pour une phrase donnée.

Cette technique a un problème important. Plus la phrase d'entrée est longue, plus l'encodeur a de la difficulté à compresser dans la représentation vectorielle toute l'information nécessaire à la traduction (Cho *et al.*, 2014a). Bahdanau *et al.* (2014) proposent une variante qui utilise un *mécanisme d'attention*. L'encodeur produit pour chaque mot d'entrée une représentation. Le décodeur utilise le mécanisme d'attention pour déterminer le poids à accorder à chaque mot d'entrée pour la génération du prochain mot de sortie. Ainsi, une représentation « dynamique » de la phrase d'entrée est construite à chaque mot de la sortie. Ce module d'attention est entrainé conjointement avec l'encodeur et le décodeur. Le problème de perte d'information dû à la « distance » entre les mots est atténué puisque le décodeur peut choisir les mots de l'entrée sur lesquels porter attention.

2.1.2. Transformer

Les RNN ont par contre une limite fondamentale importante : le modèle doit traiter séquentiellement chaque mot de l'entrée. Le calcul du plongement d'un mot dépend du plongement calculé pour le mot précédent.

Vaswani *et al.* (2017) proposent le « Transformer », un modèle séquence-à-séquence, également basé sur l'approche encodeur-décodeur, mais n'utilisant que le concept d'attention, sans celui de récurrence. Leur architecture permet à l'encodeur de calculer de façon *parallèle* le plongement de chaque mot de la phrase d'entrée. Grâce à cette parallélisation, ils obtiennent des performances égales à un RNN en moins de temps et avec moins de ressources.

1. Dans un mode autorégressif, le décodeur utilise également comme entrée le précédent mot généré.

Nous expliquons rapidement les concepts importants de l'architecture, mais invitons le lecteur à consulter l'article de Vaswani *et al.* (2017) pour les détails [2].

L'encodeur est composé de plusieurs couches identiques (six en général), chacune principalement composée d'un module d'intra-attention (« *self-attention* » en anglais) suivi d'un réseau de neurones à propagation avant (« *feed forward neural network* » en anglais) — c'est-à-dire non récurrent. Chaque couche produit, en sortie, une représentation vectorielle pour chaque mot de la phrase. La sortie d'une couche est utilisée comme entrée de la suivante. La sortie de la dernière couche est la représentation finale calculée par l'encodeur pour chaque mot.

Le module d'intra-attention, le premier module de la couche, reçoit en entrée la représentation de chaque mot calculée par la couche précédente. Si nous avons m mots, chacun représenté par un vecteur de taille d, l'entrée est donc une matrice de taille $m \times d$. Pour chaque mot, le module crée trois vecteurs appelés « *query* », « *key* » et « *value* » que nous désignerons respectivement par $Q^{(i)}$, $K^{(i)}$ et $V^{(i)}$ pour le i^e mot ($1 \leq i \leq m$). Nous expliquons plus loin comment ces vecteurs sont obtenus. En pratique, ces vecteurs sont tous de taille d. Le module va produire la nouvelle représentation du i^e mot par une moyenne pondérée des vecteurs « *value* » de chacun des m mots (incluant lui-même). Les poids de cette moyenne pondérée (les poids d'attention) sont obtenus par produit scalaire entre le vecteur « *query* » et le vecteur « *key* » de chacun des autres mots. Par exemple, le j^e poids de cette moyenne est le produit scalaire entre le vecteur $Q^{(i)}$ et le vecteur $K^{(j)}$. Le produit scalaire est normalisé entre 0 et 1 par un `softmax` sur tous les j. Les représentations obtenues sont ensuite envoyées au réseau de neurones qui produit la représentation finale pour la couche (une matrice $m \times d$).

Les auteurs de Vaswani *et al.* (2017) proposent que chaque module d'attention soit composé de plusieurs têtes parallèles (huit en général), chacune pouvant se concentrer sur certains signaux des vecteurs d'entrées. Leur sortie est ensuite combinée en une seule représentation. Pour la t^e tête, les vecteurs $Q^{(i)}$, $K^{(i)}$ et $V^{(i)}$ sont obtenus par projection de la représentation d'entrée à l'aide des matrices $W_Q^{(t)}$, $W_K^{(t)}$ et $W_V^{(t)}$ (respectivement) apprises durant l'entrainement.

Ces opérations matricielles peuvent être effectuées parallèlement sur toutes les têtes et sur tous les mots. Nous obtenons un gain majeur d'efficacité par rapport au traitement séquentiel du RNN. De plus, le chemin de la propagation de l'information entre deux mots est constant, peu importe leur distance dans la phrase source. Ceci permet au modèle de facilement traiter et apprendre les relations longue distance qui sont souvent importantes en traduction.

Le décodeur ressemble beaucoup à l'encodeur, à l'exception que dans chaque couche se trouve un deuxième module d'intra-attention. Celui-ci utilise la sortie de la couche précédente comme vecteur « *query* », mais la sortie de *l'encodeur* comme valeur des vecteurs « *key* » et

2. Nous proposons également la lecture de l'excellent article « The Annotated Transformer » (`http://nlp.seas.harvard.edu/2018/04/03/attention.html`)

« *value* ». Ceci permet de calculer une attention entre les mots générés et les mots d'entrée. Notons également que le décodeur ne porte attention qu'aux mots déjà générés dans la sortie, et non à tous les mots (passés et futurs) comme le fait l'encodeur. Une transformation linéaire suivie d'un `softmax` est finalement appliquée sur la sortie de la dernière couche pour obtenir une probabilité pour chaque mot du vocabulaire. Le modèle est entrainé comme les modèles à RNN, de façon supervisée et en maximisant la log-probabilité de l'hypothèse de traduction.

L'efficacité du Transformer en a fait l'architecture de traduction la plus populaire en TAN (Barrault *et al.*, 2020; Fraser, 2020) et nous l'utilisons pour le modèle que nous proposons.

2.2. Transfert d'apprentissage pour l'adaptation de domaine

Nous nous intéressons à la problématique suivante : nous souhaitons créer, rapidement, simplement et efficacement, un modèle de TAN spécialisé dans un domaine cible pour lequel nous n'avons accès qu'à un petit corpus. Nous appellerons ce corpus le « corpus cible ». Ce corpus est généralement trop petit pour permettre de l'utiliser seul pour l'entrainement d'un modèle spécialisé. Nous avons par contre accès à un grand corpus de textes parallèles (bilingues et alignés) composés d'un ensemble de domaines variés, différents du domaine cible. Nous appellerons ce corpus le « corpus général ». Comment profiter du corpus général pour créer un modèle de traduction efficace pour le domaine cible ?

Nous distinguons deux cas : soit le corpus du domaine cible est composé de phrases parallèles, soit il est uniquement composé de phrases unilingues, dans la langue source ou cible. Nous parlons d'apprentissage supervisé dans le premier cas, et d'apprentissage non supervisé dans le deuxième (à noter que certains auteurs parlent plutôt d'apprentissage semi-supervisé voir, par exemple, Currey *et al.*, 2017; Jin *et al.*, 2020). La création d'un modèle spécialisé est plus aisée en apprentissage supervisé, mais, comme nous avons vu à la section 1.4.1, les domaines avec corpus parallèles sont rares. Il est, par contre, souvent relativement simple de construire un corpus unilingue pour un domaine. L'apprentissage non supervisé, bien que plus complexe, est donc souvent la meilleure approche, voire la seule possible.

Le modèle que nous proposons au chapitre 3 est un modèle d'apprentissage non supervisé, mais nous survolons ici les deux situations pour mieux mettre en contexte les approches que nous avons choisies pour notre modèle.

2.2.1. Transfert d'apprentissage supervisé

Une solution « évidente » à notre problème de corpus trop petit serait d'augmenter sa taille. Certains auteurs émettent l'hypothèse que l'on peut retrouver à même le corpus général des

phrases similaires à notre domaine cible, même si celui-ci n'y est pas explicitement représenté. Ils proposent d'aller y extraire ces phrases pour **augmenter le corpus cible**. Différentes approches proposent différentes mesures de similarités.

Moore et Lewis (2010) utilisent des modèles de langue afin de calculer un score de similarité entre le domaine cible et chaque phrase du corpus général. Un modèle de langue (« *language model* » en anglais) est un modèle statistique qui assigne une probabilité à une phrase. La probabilité d'une phrase étant la probabilité d'une telle séquence de mots selon son domaine d'entrainement. À partir de cette distribution de probabilité, il est possible de calculer l'entropie croisée d'une séquence de mots. Moore et Lewis (2010) entrainent un modèle de langue sur le corpus général et un deuxième sur le corpus cible. En calculant, pour chaque phrase du corpus général, la différence des entropies croisées, ils obtiennent un score de similarité au domaine cible, ce qui leur permet de sélectionner des phrases pour le corpus cible. Axelrod *et al.* (2011) adaptent cette méthode aux corpus bilingues en calculant un score basé sur la différence d'entropie des phrases sources et cibles. Duh *et al.* (2013) adapte la technique pour les modèles de langue basés sur les réseaux de neurones.

Chinea-Rios *et al.* (2017) proposent plutôt d'utiliser une représentation vectorielle, ou plongement (« *embedding* » en anglais), de la phrase pour déterminer sa similarité avec le domaine cible. Une représentation pour chaque phrase est calculée à l'aide d'une approche basée sur le « *Continuous Skip-gram Model* » de Mikolov *et al.* (2013). Les auteurs émettent l'hypothèse que les vecteurs obtenus pour les phrases du domaine cible forment un amas dans l'espace vectoriel et proposent d'utiliser leur centroïde comme représentation du domaine cible. Ils sélectionnent ensuite, parmi le corpus général, les phrases qui se retrouvent à l'intérieur d'une hypersphère centrée sur le domaine. Wang *et al.* (2017b) calculent plutôt la représentation d'une phrase à partir des états cachés produits par l'encodeur d'un modèle de TAN préentrainé. Le domaine est encore une fois représenté par le centroïde de ses phrases. La distance euclidienne est utilisée pour déterminer les phrases du corpus général les plus proches du domaine cible. Contrairement à Chinea-Rios *et al.* (2017) qui n'utilisent que les textes dans la langue source, Wang *et al.* (2017b) utilisent deux modèles de traduction pour calculer une similarité bilingue.

Ces stratégies font ressortir la source d'efficacité du transfert d'apprentissage : parmi le corpus général se trouvent des informations qui peuvent être utilisées pour faciliter l'apprentissage de la tâche de traduction sur le domaine cible.

Cependant, les stratégies précédentes cherchent à extraire du corpus général un *sous-ensemble* de phrases semblables à notre domaine pour entrainer un modèle. Nous savons que les modèles basés sur les réseaux de neurones nécessitent de grands corpus d'entrainement. Sélectionner seulement certaines phrases et rejeter les autres nous prive d'une information utile pour

l'apprentissage de la tâche de traduction. Il serait plus judicieux de profiter de toutes les données disponibles.

Wang *et al.* (2017c) proposent d'utiliser la technique de **pondération d'instances** (« *instance weighting* » en anglais) pour la traduction automatique neuronale. Tout le corpus (général et cible) est utilisé pour l'entrainement, mais le modèle accorde plus ou moins de poids à chaque phrase selon sa similarité au domaine cible. Wang *et al.* (2017c) calculent le poids selon la technique d'entropie croisée bilingue proposée par Axelrod *et al.* (2011) et l'utilisent dans le calcul de la fonction objective de la tâche de traduction. Ils tentent également une expérience simple où ils s'assurent d'un certain ratio de phrases du domaine cible dans chaque mini-lot (« *mini-batch* » en anglais) d'entrainement. Chen *et al.* (2017) entrainent plutôt un classificateur qui, à partir des états cachés de l'encodeur, calcule une probabilité d'appartenance de la phrase d'entrée au domaine cible. Cette probabilité est ensuite utilisée pour pondérer, pour chaque phrase, la fonction d'erreur lors de l'entrainement. L'avantage de cette méthode est que le classificateur qui calcule la pondération est entrainé en parallèle du modèle et sur le même corpus. De plus, un unique modèle est utilisé pour la pondération et la traduction, ce qui permet un transfert de connaissances entre les deux tâches.

La pondération d'instances réussit à produire des modèles spécialisés en valorisant, lors de l'entrainement, les phrases semblables au domaine cible. Néanmoins, l'entrainement est ralenti par le processus de pondération de chaque phrase du corpus qui doit être exécuté à chaque nouveau domaine d'intérêt, complexifiant et ralentissant la création d'un modèle.

Pour éviter cette étape de préentrainement, pourrait-on simplement entrainer un modèle sur la combinaison simple des deux corpus, sans aucune pondération ? Freitag et Al-Onaizan (2016) soulignent le problème avec cette approche : le domaine cible se retrouve très faiblement représenté dans la distribution des phrases des corpus fusionnés. Un modèle entrainé sur cet ensemble risque d'être biaisé et peu performant sur le domaine d'intérêt.

Luong et Manning (2015) proposent la **méthode d'affinage** (« *fine-tuning* » en anglais), une approche permettant de s'entrainer sur les deux corpus, mais de façon efficace et sans risquer de noyer le domaine d'intérêt dans le corpus général. La technique considère la tâche de traduction sur le domaine d'intérêt comme une combinaison de deux sous-tâches qui peuvent être apprises séquentiellement : une tâche de traduction générique et une tâche de spécialisation sur le domaine. Un modèle de TAN est premièrement entrainé sur le corpus général jusqu'à convergence. Son entrainement est ensuite poursuivi, dans une deuxième étape, sur le corpus cible pour le spécialiser. Cette technique simple et très efficace est, encore aujourd'hui, souvent utilisée comme modèle de base. Elle permet de créer des modèles spécialisés et compétitifs face à un modèle entrainé uniquement sur le domaine cible, et en un temps réduit (Servan *et al.*, 2016).

La méthode est par contre sensible à deux problèmes : le surapprentissage et l'oubli catastrophique (« *catastrophic forgetting* » en anglais). Le surapprentissage est causé par la faible taille du corpus cible. Le modèle peut rapidement apprendre « par-cœur » l'ensemble d'entrainement et performer faiblement sur de nouvelles données. Le problème est, de plus, exacerbé par les capacités toujours plus grandes des nouvelles architectures de modèle. L'oubli catastrophique (McCloskey et Cohen, 1989) survient lorsqu'un modèle, qui obtenait de bonnes performances sur un domaine A, augmente, par affinage sur un domaine B, ses performances sur le domaine B, mais au détriment de celles sur le domaine A. En d'autres mots, l'affinage fait « oublier » au modèle des compétences acquises lors de son entrainement initial.

Divers auteurs proposent des solutions pour limiter l'oubli catastrophique. Limiter le nombre d'époques d'affinage est une technique simple, mais elle se fait au détriment des performances sur le domaine cible (Saunders, 2021). Freitag et Al-Onaizan (2016) proposent d'utiliser, au moment du décodage, un ensemble composé du modèle affiné et du modèle non affiné. Dakwale et Monz (2017) démontrent par contre que les performances se détériorent quand même sur les domaines d'entrainement du modèle non affiné. Ils proposent une technique de « *Knowledge Distillation* » (Hinton *et al.*, 2015) afin de préserver, durant l'affinage, la distribution du corpus général apprise par le modèle.

Limiter l'oubli catastrophique permet de contrôler la dégradation sur les domaines du corpus général, mais a pour effet secondaire de limiter les performances du modèle sur le domaine cible. Nous pouvons complètement éviter l'oubli catastrophique en affinant un nouveau modèle sur chaque domaine. Cette stratégie devient par contre rapidement couteuse en temps et en ressources avec l'augmentation des domaines à supporter.

Bapna *et al.* (2019) proposent d'utiliser des « **adaptateurs** » permettant d'affiner seulement une partie du modèle tout en préservant les connaissances acquises sur les domaines du corpus général. Puisque le temps d'affinage est réduit, il devient plus rentable d'affiner un modèle sur chaque domaine. Dans cette technique, de petits réseaux de neurones, les « adaptateurs », sont insérés entre les couches d'un modèle de TAN préentrainé sur le corpus général. Au moment de l'affinage, seules ces nouvelles couches sont entrainées, les autres paramètres restant fixes. Le nombre de paramètres ajoutés par ces adaptateurs est grandement inférieur au nombre de paramètres total du réseau, permettant un affinage rapide et un coût en mémoire réduit. Un ensemble d'adaptateurs différent peut être entrainé pour chaque domaine cible, et le modèle sans adaptateur peut être utilisé pour tout autre domaine.

Le problème de l'oubli catastrophique est contourné, mais l'autre problème important de l'affinage reste : le risque de surapprentissage si l'ensemble d'entrainement du domaine cible est limité. Nous avons souligné la difficulté d'obtenir de grands corpus pour la majorité des domaines.

La stratégie du modèle **multidomaines** est une technique de transfert d'apprentissage différente qui offre une solution au surapprentissage des petits domaines tout en évitant l'oubli catastrophique. L'intuition est que les informations présentes dans les données d'entrainement d'un gros domaine peuvent être utilisées pour favoriser l'apprentissage d'un autre domaine qui lui serait similaire, même si son corpus est petit. Entrainer un modèle sur plusieurs domaines en même temps permet un transfert de connaissances entre les domaines, sans subir de dégradation de performances (Tars et Fishel, 2018). L'ensemble d'entrainement se compose d'une multitude de domaines et une variable catégorielle permet au modèle de déterminer le domaine de la phrase en cours.

L'idée a déjà démontré ses possibilités dans les modèles multilingues. Johnson *et al.* (2017) ont créé un modèle unique qui peut traduire entre plusieurs paires de langues. Au moment de la traduction d'une phrase, un code de langue (par exemple « `<2en>` ») est fourni au modèle pour indiquer la langue de traduction souhaitée. Ils ont remarqué que ce transfert de connaissance a permis au modèle d'améliorer son apprentissage sur les paires de langues les plus pauvres sans dégrader celles sur les paires les plus riches.

Nous soulignons l'importance de fournir au modèle l'information sur le domaine de la phrase à traduire (ou sur la langue cible, dans le cas de Johnson *et al.* (2017)). Simplement fusionner les corpus de tous les domaines, sans aucune information permettant de les distinguer, n'est pas suffisant pour entrainer un modèle multidomaines et peut même dégrader les performances sur les domaines individuels (Britz *et al.*, 2017). Différentes stratégies sont proposées pour le passage cette information.

Kobus *et al.* (2016) entrainent un modèle sur la fusion des corpus, mais ajoutent à la fin de chaque phrase source un mot-étiquette indiquant le domaine (par exemple `@MED@`). Ils tentent aussi une expérience en fusionnant le plongement du mot-étiquette au plongement de chaque mot de la phrase d'entrée. Ils démontrent que leur modèle obtient de meilleurs résultats sur les petits domaines qu'un modèle entrainé uniquement sur ceux-ci. En interchangeant les mots-étiquettes entre les domaines, ils démontrent aussi que certains domaines sont plus « proches » que d'autres. Chu *et al.* (2017) proposent une technique appelée *mixed fine-tuning* mélangeant les techniques d'affinage et d'ajout d'un mot-étiquette. Un modèle est premièrement entrainé sur le corpus général dans lequel chaque phrase est précédée d'un mot-étiquette représentant son domaine, comme dans Johnson *et al.* (2017). L'entrainement du modèle est ensuite poursuivi sur un mélange de données multidomaines et du domaine cible, où ce dernier peut être artificiellement surreprésenté. Le modèle obtenu peut, si désiré, être ensuite affiné uniquement sur le domaine cible. Les auteurs explorent aussi dans Chu *et al.* (2018) l'utilisation d'un corpus généré par rétrotraduction, une méthode que nous verrons dans la section 2.2.2. Tars et Fishel (2018) adaptent deux méthodes multilingues à la création d'un modèle multidomaines. Inspiré premièrement de Johnson *et al.* (2017)

ils précèdent chaque phrase d'un mot-étiquette du domaine. Ils expérimentent ensuite une technique inspirée de Östling et Tiedemann (2016) où le plongement du domaine est fusionné au plongement de chaque mot dans l'encodeur. Le calcul des plongements est fait à l'aide de Nematus (Sennrich *et al.*, 2017). Plus récemment, Stergiadis *et al.* (2021) proposent de fournir au modèle une information multidimensionnelle. En plus du domaine, plusieurs autres méta-informations pourraient être fournies au modèle, comme la langue, ou la source de la phrase. Ils utilisent également le principe du mot-étiquette rajouté à la phrase source pour passer cette information multidimensionnelle.

La technique du modèle multidomaines comble différents aspects de notre objectif : un unique modèle est adapté à plusieurs domaines, permettant de l'utiliser immédiatement sur n'importe lequel des domaines d'entrainement ; l'entrainement est simple et relativement rapide (le ratio du temps d'entrainement divisé par le nombre de domaines supportés est faible) ; finalement, il peut être efficace sur les petits domaines, sans surapprentissage, grâce au transfert des connaissances acquises sur les plus gros domaines. Le modèle que nous proposons est directement inspiré de la technique multidomaines.

Soulignons que l'information sur le domaine est fournie au modèle depuis l'extérieur, en précédant manuellement, par exemple, chaque phrase d'un mot-étiquette. D'autres auteurs proposent plutôt des architectures où le modèle est entrainé à extraire lui-même l'information sur le domaine uniquement à partir de la phrase d'entrée.

Britz *et al.* (2017) proposent d'ajouter un autre objectif au modèle, en plus de la fonction objective de traduction. Une première proposition ajoute un classificateur à l'encodeur qui doit déterminer le domaine de la phrase. Ceci force l'encodeur à inclure dans la représentation qu'il calcule une information spécifique au domaine. Une deuxième proposition force plutôt le décodeur à inclure une information générique au domaine. Une troisième proposition demande au modèle de prédire le domaine par le décodeur, forçant plutôt ce dernier à distinguer le domaine. Dans la même veine, Zeng *et al.* (2018) utilisent dans l'encodeur un classificateur de domaine ainsi qu'un classificateur antagoniste (« *adversarial* ») afin d'extraire de la phrase d'entrée une représentation spécifique de l'information du domaine et une représentation générique au domaine. Ces représentations sont ensuite chacune intégrées avec la représentation calculée par l'encodeur et fournies au décodeur. Le décodeur doit également accomplir une tâche de classification. Finalement, notons Pham *et al.* (2020) qui utilisent les adaptateurs (Bapna *et al.*, 2019) dans le contexte de la traduction multidomaines. Lors de l'entrainement, une composante supplémentaire est entrainée pour décider d'activer ou non les différentes couches d'adaptateurs sur une base mot-à-mot.

L'intérêt de ces techniques repose sur leur capacité à construire une représentation interne du domaine. Bien que ces modèles soient entrainés sur un nombre fini de domaines et conçus pour se spécialiser sur un ou des domaines précis, rien ne les empêche de calculer une représentation

à partir d'une phrase issue de n'importe quel domaine, même un qui n'aurait jamais été vu à l'entrainement. Ceci ouvre la porte à un modèle qui pourrait être adaptable à n'importe quel domaine. Le modèle que nous proposons au chapitre 3 propose des stratégies pour construire une telle représentation pour tout domaine.

Ces techniques calculent cependant la représentation à partir d'une phrase unique. L'information sur le domaine peut y être bruitée et une information plus claire pourrait sans doute être retirée d'un plus grand ensemble de phrases extraites du même domaine. Nous présentons à la section §2.3 diverses stratégies pour construire une représentation à partir d'un plus grand ensemble.

Nous terminons cette section sur un aspect de l'objectif qui a peu été couvert : l'**adaptation rapide** à un nouveau domaine. En apprentissage supervisé, nous ne citerons que Sharaf *et al.* (2020), mais d'autres techniques seront présentées dans la section sur l'apprentissage non supervisé.

Sharaf *et al.* (2020) utilisent les concepts du méta-apprentissage pour créer un modèle qui pourra s'adapter rapidement et efficacement à tout nouveau domaine. Un algorithme de méta-apprentissage n'a pas pour objectif de créer un modèle entrainé pour un domaine, mais plutôt de créer une *initialisation* de modèle, prête pour un entrainement rapide et efficace sur n'importe quel domaine. Les auteurs proposent un algorithme d'apprentissage, basé sur la technique MAML (Finn *et al.*, 2017), qui simule des tâches variées d'adaptation de domaine et entraine les paramètres afin de réduire l'erreur obtenue à chaque simulation. Cette architecture permet une adaptation rapide à un nouveau domaine avec souvent seulement quelques centaines de phrases.

Malgré la rapidité de l'adaptation, la stratégie requiert quand même un entrainement pour tout nouveau domaine, ce qui nécessite des ressources en temps et en technologies. Soulignons également que, comme la technique d'affinage, le modèle produit n'est spécialisé que sur un seul domaine.

2.2.2. Transfert d'apprentissage non supervisé

Il est rare d'avoir accès à des corpus d'entrainement parallèles pour un domaine d'intérêt. Un corpus unilingue est généralement beaucoup plus simple à créer. Un modèle qui peut se spécialiser sur un domaine d'intérêt uniquement à partir de phrases unilingues (apprentissage non supervisé) a un avantage évident par rapport à une stratégie supervisée nécessitant des phrases bilingues. Différentes stratégies d'adaptation de domaine non supervisé ont été proposées.

Gülçehre *et al.* (2015) proposent d'entrainer un modèle de langue, extérieur au modèle de traduction, sur le corpus unilingue [3] du domaine cible. Ce modèle de langue est ensuite utilisé pour guider le modèle de traduction. Les auteurs explorent deux techniques d'intégration du modèle de langue à celui de traduction : une version superficielle où il est utilisé pour repondérer les hypothèses produites par le modèle de TAN, et une version profonde où il est fusionné avec le décodeur. La stratégie proposée par les auteurs nécessite par contre beaucoup de données et requiert l'entrainement de deux modèles séparés. Domhan et Hieber (2017) proposent une méthode similaire où les modèles de traduction et de langue sont entrainés conjointement.

Une autre stratégie consiste à transformer notre ensemble de données unilingue du domaine d'intérêt en un ensemble bilingue afin d'utiliser les stratégies supervisées présentées à la section 2.2.1. Les données manquantes peuvent être générées synthétiquement de diverses façons.

Sennrich *et al.* (2015) assigne simplement une phrase nulle (par exemple « NULL ») comme phrase source de chaque phrase cible du corpus unilingue et les intègrent dans le corpus général. Les auteurs remarquent par contre qu'ils créent ainsi un modèle multitâches où sont soit entrainées les capacités de traduction, soit les capacités linguistiques, selon la phrase d'entrée. Les connaissances acquises dans chaque tâche ne sont pas partagées avec l'autre. Currey *et al.* (2017) copient directement la phrase cible vers la source avant de les fusionner avec le corpus général. Ils réussissent à améliorer l'entrainement du décodeur par rapport au domaine cible sans recréer le problème du modèle multitâches. Hu *et al.* (2019) proposent un modèle qui construit, à partir de corpus bilingues mais non-alignés, un lexique bilingue spécialisé dans le domaine cible. Ils génèrent ensuite une phrase source pour chaque phrase cible en traduisant cette dernière mot-à-mot à l'aide du lexique.

Sennrich *et al.* (2015) proposent également la technique de rétrotraduction (« *back-translation* ») pour générer des phrases sources à partir d'un corpus unilingue dans la langue cible. Un modèle de traduction est premièrement entrainé dans la direction linguistique *inverse* cible→source. Ce modèle est ensuite utilisé pour générer des phrases sources en traduisant chaque phrase cible. Les données bilingues ainsi générées sont ajoutées au corpus général pour entrainer un modèle de traduction source→cible. Malgré la simplicité de la méthode, elle a plusieurs fois démontré son efficacité, ses performances dépassant parfois même celles d'autres modèles plus modernes (Edunov *et al.*, 2018; Bojar *et al.*, 2018; Barrault *et al.*, 2019; Hu *et al.*, 2019; Jin *et al.*, 2020). La technique a par contre une faiblesse importante : sa performance dépend directement de la qualité du modèle de rétrotraduction. En particulier, si l'on souhaite rétrotraduire les phrases d'un domaine particulier, mais que le modèle n'a pas été entrainé sur celui-ci, la qualité de ses traductions pourrait être limitée.

3. Le corpus est unilingue dans la langue cible.

Dou *et al.* (2020) proposent une architecture où le modèle de rétrotraduction est entrainé conjointement avec le modèle de traduction sur le domaine cible. De plus, les phrases à rétrotraduire sont ordonnées de façon à débuter avec les phrases les plus « faciles » pour tranquillement amener le modèle à produire les traductions plus spécifiques au domaine cible.

La génération de données synthétiques permet donc de transformer une tâche d'apprentissage non supervisée en une tâche supervisée. Cependant, cette étape peut avoir un coût important en temps et en ressources, coût qui doit être ajouté à celui du processus d'entrainement.

D'autres auteurs proposent plutôt des méthodes entièrement, ou partiellement, non supervisées, centrées sur l'architecture du modèle et le processus d'entrainement, plutôt qu'uniquement sur la génération de données synthétiques.

Afin d'apprendre simultanément la tâche de traduction et de spécialisation, la fonction objective du modèle est souvent divisée en deux ou plusieurs objectifs. L'objectif de traduction s'y trouve toujours, mais un objectif de modèle de langue, unilingue ou bilingue, est souvent rajouté afin d'adapter l'encodeur et le décodeur au domaine cible. Dou *et al.* (2019) proposent un modèle qui divise la représentation de la phrase d'entrée en trois composantes : une représentation du domaine, une représentation de la tâche et une représentation commune à tous les domaines et les tâches. Ils obtiennent ainsi un modèle multidomaines et multitâches. Le modèle est entrainé avec deux fonctions objectives : une de traduction et une de modèle de langue. Jin *et al.* (2020) utilisent un modèle de TAN préentrainé qu'ils entrainent ensuite avec trois objectifs : un de traduction supervisée, un de modèle de langue et un de rétrotraduction où le modèle doit reconstruire la phrase source. Leur modèle est entrainé dans les deux langues en même temps, permettant d'adapter l'encodeur et le décodeur. Mahdieh *et al.* (2020) proposent une architecture semblable à Jin *et al.* (2020), mais capable de s'adapter rapidement à de nouveaux domaines. Leur technique préentraine un modèle de traduction à l'aide du corpus général bilingue et d'une tâche de modèle de langue sur des corpus bilingues, mais non parallèles. Cette dernière tâche utilise une technique appelée MASS (Song *et al.*, 2019), semblable à BERT (Devlin *et al.*, 2018), où le modèle s'entraine sur un corpus bilingue, mais non parallèle, le forçant à créer des représentations indépendantes de la langue. Ils obtiennent un modèle préentrainé qui peut ensuite rapidement s'adapter par affinage sur un corpus unilingue en langue source.

Ces stratégies de division de l'objectif en sous-objectifs rappellent celle de l'affinage où la tâche de traduction est divisée en deux principales tâches, permettant à l'algorithme d'apprentissage de spécialiser plusieurs aspects du modèle en même temps. Par contre, un des problèmes rencontrés dans l'affinage se présente également ici : le modèle reste spécialisé dans un seul domaine, un réentrainement étant nécessaire pour chaque nouveau domaine.

Pour éviter ce problème d'un modèle limité à un domaine, Li *et al.* (2016) proposent une stratégie centrée sur la phrase plutôt que sur le domaine. Au lieu d'avoir un modèle adapté à un domaine, il est adapté pour chaque phrase. Le modèle est d'abord préentrainé sur le corpus général. Au moment de l'inférence, une mesure de similarité, inspirée de Mikolov *et al.* (2013), est utilisée pour trouver dans le corpus général un mini-ensemble composé des phrases les plus proches de la phrase d'entrée. Le modèle est alors affiné sur ce mini-ensemble pour obtenir un modèle adapté à la phrase. Le processus est repris pour chaque nouvelle phrase. Farajian *et al.* (2017) proposent une extension où les hyperparamètres de l'affinage sont déterminés dynamiquement selon la similarité entre la phrase d'entrée et chaque phrase sélectionnée. Zhang *et al.* (2018) proposent également une méthode qui récupère du corpus général des phrases similaires à la phrase d'entrée. L'intérêt de leur technique est que le modèle de traduction n'a pas à être réentrainé ni affiné. Leur technique repose sur la recherche de « morceaux de traduction », des n-grammes bilingues dans le corpus général similaires avec les n-grammes de la phrase d'entrée. Les hypothèses du modèle de traduction sont ensuite pondérées selon la présence de ces morceaux de traduction.

Ces modèles ont un avantage théorique important sur les autres : en travaillant uniquement au niveau de la phrase, ils ne sont plus astreints à un système catégorique de domaines. Ils ne sont plus limités à faire un choix, même lorsqu'aucun domaine ne correspond. Ils peuvent être utilisés sur n'importe quel domaine, même s'il n'a pas été vu à l'entrainement. Ils souffrent cependant d'un problème d'efficacité évident, nécessitant une phase d'entrainement pour chaque phrase à traduire.

Nous avons annoncé, à la section 2.2.1, que le modèle que nous proposons au prochain chapitre sera basé sur la stratégie du modèle multidomaines. Le modèle ne sera pas restreint, lui non plus, à traduire un ensemble limité de domaines. Il sera également en mesure de s'adapter de façon plus efficace.

2.3. Utilisation du contexte de la phrase

Nous avons présenté à la section 2.2.1 une famille de modèles qui produisent une représentation interne du domaine uniquement à partir de la phrase d'entrée. Nous avons souligné que leur procédé pouvait être appliqué à n'importe quel domaine et que ceci ouvre la porte à des modèles capables de s'adapter à des domaines qui n'auraient jamais été vus à l'entrainement.

Cependant, les modèles de base de TAN traduisent chaque phrase indépendamment, une à la fois, sans accès aux autres phrases. Les modèles qui construisent une représentation du domaine doivent donc le faire uniquement à partir de la phrase d'entrée. Nous avons souligné que l'information sur le domaine dans une seule phrase est probablement très bruitée et une information plus claire pourrait sans doute être extraite si un plus grand ensemble de phrases

était utilisé. Le modèle que nous proposons cherche à créer une représentation du domaine à partir d'un tel ensemble.

L'importance de l'accès à un contexte plus large de la phrase va d'ailleurs au-delà de la construction d'une représentation du domaine. Il s'agit d'une information utile pour le choix des bons mots, des bons pronoms, pour la résolution des coréférences, pour la cohésion lexicale, etc.

Comment fournir un plus large contexte à un modèle qui traduit phrase par phrase ? Différentes méthodes ont été suggérées.

Tiedemann et Scherrer (2017) expérimentent un modèle où la phrase d'entrée est la concaténation de la phrase précédente et de la phrase à traduire. Jean *et al.* (2017) utilisent k encodeurs pour créer une représentation pour chacune des k phrases sources précédentes. À chaque étape de génération de mot, un modèle d'attention spécial, entrainé avec le modèle, construit une représentation unique à partir des k précédentes. Ceci permet au décodeur d'avoir accès au contexte précédent de la phrase pour la génération du prochain mot. Vu le lien linéaire entre k et la taille du modèle, ils limitent leurs expériences à $k = 1$. Wang *et al.* (2017a) utilisent une hiérarchie de RNN pour calculer une représentation des phrases sources précédentes. Un premier RNN calcule une représentation pour chacune des k phrases précédentes. Cette séquence de représentations est ensuite envoyée dans un deuxième RNN qui calcule une représentation globale du contexte précédent la phrase en cours. Cette représentation est ensuite utilisée par le décodeur. Miculicich *et al.* (2018) font remarquer que cette hiérarchie de RNN ne permet pas au modèle de choisir le niveau d'attention qu'il porte à chacune des phrases ni chacun de leurs mots. Ils proposent donc un modèle utilisant un réseau d'attention hiérarchique. Une fonction d'attention multitêtes, semblable à celle proposée par Vaswani *et al.* (2017), calcule une représentation pour chacune des k phrases précédentes. Celles-ci sont utilisées par une autre fonction d'attention pour calculer une représentation unique. Ces attentions peuvent être appliquées sur les phrases sources, les phrases cibles, et être utilisées dans l'encodeur ou le décodeur. Tu *et al.* (2018) soulignent que les méthodes de Jean *et al.* (2017) et Wang *et al.* (2017a) se limitent aux phrases sources et qu'ils ont un coût computationnel élevé. Ils proposent un modèle rehaussé d'une « cache continue » qui garde en mémoire les dernières représentations calculées. Le modèle d'attention peut ainsi, à faible coût, utiliser plusieurs représentations passées en interrogeant la cache.

Macé et Servan (2019) élargissent le contexte à tout le document source d'où la phrase a été extraite, et non plus aux quelques phrases précédentes. Ils argumentent que l'information nécessaire à la traduction peut se trouver plusieurs phrases avant, et même après la phrase en cours. Une approche simple, appelée *SWEM-aver*, proposée par Shen *et al.* (2018), calcule le plongement du document en faisant la moyenne du plongement de tous ses mots. Le plongement des mots sont extraits depuis ceux appris par un Transformer (Vaswani *et al.*,

2017) préentrainé. Ils précèdent ensuite chaque phrase source d'un mot-étiquette indiquant le document d'origine. Le modèle est mis à jour pour que le plongement de cette étiquette soit le plongement du document. Jiang *et al.* (2020) utilisent dans leur modèle le plongement du document source en plus d'un plongement calculé à partir des phrases sources autour de la phrase à traduire. Ces deux plongements sont ensuite ajoutés en début de séquence des plongements des mots de la phrase source.

Les articles précédents définissent le contexte de la phrase en cours à différents niveaux. Certains ne considèrent que quelques mots ou une phrase précédente, d'autres regardent quelques phrases voisines, et d'autres élargissent le contexte à tout le document. À notre connaissance, nous n'avons pas vu d'articles qui étendent le contexte à tout le domaine.

Le modèle que nous proposons est basé sur l'idée de Macé et Servan (2019). Mais là où leur stratégie est appliquée au niveau du document, nous l'étendons au niveau du domaine. Calculer une représentation du domaine offre ses propres défis, en particulier parce que le domaine est un ensemble potentiellement infini de phrases, contrairement à un document. Nous proposons au chapitre 3 diverses stratégies pour calculer la représentation du domaine. La méthode de Macé et Servan (2019) combinée au modèle multidomaines permettra de produire un modèle capable de s'adapter, de façon non supervisée et sans entrainement supplémentaire, à tout nouveau domaine, même s'il n'a jamais été vu à l'entrainement.

Chapitre 3

Approche proposée

3.1. Le modèle

Les méthodes que nous avons survolées au chapitre 2 ne sont généralement pas applicables de façon réaliste dans l'industrie de la traduction. Elles demandent souvent un entrainement couteux pour chaque nouveau domaine, et requièrent un corpus de dizaines, voire de centaines, de milliers de phrases, parfois bilingues, dans le domaine cible pour réussir l'adaptation. Une organisation de traduction aura rarement les ressources nécessaires pour entrainer un tel modèle et les textes à traduire sont habituellement limités à quelques dizaines ou centaines de phrases, dans la langue source uniquement.

La solution que nous proposons produit un modèle de TAN *préentrainé*, qui peut s'adapter à tout nouveau domaine *sans entrainement* et uniquement à partir d'un *petit ensemble* de textes en langue source.

Notre méthode est inspirée de Macé et Servan (2019), ainsi que des approches exposées dans Chinea-Rios *et al.* (2017) et Tars et Fishel (2018). Un Transformer (Vaswani *et al.*, 2017) est entrainé sur une tâche de traduction supervisée, comme à la section §2.1. Mais, en plus de la phrase d'entrée, le modèle reçoit une représentation vectorielle, ou *plongement*, du domaine d'où provient la phrase. Ce plongement est calculé à partir d'un ensemble de phrases en *langue source* du domaine d'intérêt. Différentes techniques pour calculer le plongement sont possibles et nous en expérimentons plusieurs (voir section §3.2). Pour l'entrainement, nous utilisons un corpus parallèle composé de plusieurs domaines variés (voir chapitre 4). Au moment du test, pour un nouveau domaine non vu à l'entrainement, un plongement est calculé à partir de ses phrases en langue source. Le plongement est ensuite passé au modèle lors de la traduction de chacune des phrases. Le même plongement pourra être utilisé pour de futures phrases du même domaine. Nous soulignons qu'aucun entrainement supplémentaire

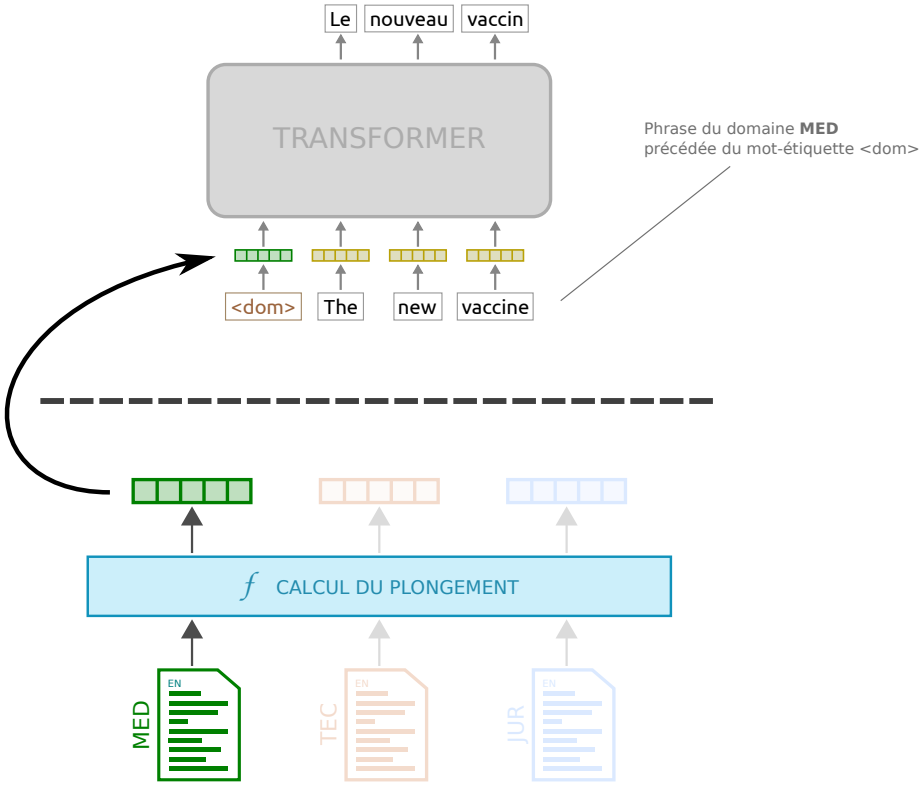

Figure 3.1. Le modèle que nous proposons. Dans cette illustration, le modèle de traduction (TRANSFORMER) doit traduire une phrase du domaine `MED` (« *The new vaccine* »). Le mot-étiquette `<dom>` est ajouté en début de phrase et son plongement est défini à celui du domaine `MED`. Celui-ci est calculé à partir des textes du domaine par une des techniques de plongement. Si la phrase avait plutôt été du domaine `TEC`, le plongement calculé pour ce domaine aurait été utilisé.

n'est effectué au moment du test. La seule étape de préparation est la construction du plongement pour le nouveau domaine. La figure 3.1 présente graphiquement notre approche.

L'intuition derrière notre modèle repose sur un concept important exploré au chapitre chapitre 2. Même si un domaine n'est pas explicitement représenté dans un corpus, ce dernier contient une information qui peut être exploitée pour l'adaptation à ce nouveau domaine. Nous pouvons présumer que plus le corpus est grand, et plus les textes qui le composent sont variés, plus l'information qui s'y trouve est riche. Nous déduisons de cette observation qu'il n'y a pas de limite claire entre les domaines, qu'il y a chevauchement, et que l'espace des domaines n'est pas discret, mais continu. Nous proposons une interprétation où un domaine est vu comme une composition de « signaux ». Différents domaines ont des signatures de signaux différentes, mais ils peuvent partager certains signaux. Notre hypothèse est qu'un modèle de traduction qui aurait appris à reconnaitre les signaux des domaines pourra plus facilement s'adapter à un nouveau domaine si on lui fournit sa « signature ». En effet, le

modèle pourrait reconnaitre dans ces signaux certains pour lesquels il a déjà acquis des connaissances.

L'objectif de la technique de plongement de domaine est alors d'incorporer le plus d'informations possibles sur ces « signaux » dans la représentation vectorielle qu'elle produit. Nous expérimentons avec diverses techniques de plongement que nous détaillons à la section §3.2.

Mais une fois le plongement calculé, comment le passer au modèle ? Différentes stratégies existent pour fournir au modèle une telle méta-information. Certains auteurs additionnent le vecteur de méta-information au plongement de chaque mot (voir, par exemple, Vaswani *et al.*, 2017; Devlin *et al.*, 2018; Lample et Conneau, 2019; Jin *et al.*, 2020). D'autres concatènent les deux vecteurs (voir, par exemple, Kobus *et al.*, 2016; Östling et Tiedemann, 2016; Tars et Fishel, 2018), avec l'avantage qu'ils n'ont pas à avoir la même taille. Une autre option est d'intégrer le vecteur de méta-information au sein de la phrase d'entrée en tant que mot supplémentaire. Selon cette stratégie, un mot spécial, que nous appellerons « mot-étiquette », est rajouté en début ou en fin de phrase (plusieurs mots-étiquettes peuvent aussi être utilisés dans un même modèle). Le mot-étiquette ne doit pas être un mot déjà présent dans le vocabulaire du modèle. Comme tous les autres mots, celui-ci a un plongement vectoriel associé. L'idée est de remplacer ce plongement par le vecteur de méta-information. Cette technique est, entre autres, utilisée dans Kobus *et al.* (2016); Sennrich *et al.* (2016a); Yamagishi *et al.* (2016); Johnson *et al.* (2017); Chu *et al.* (2017); Tars et Fishel (2018); Macé et Servan (2019); Jiang *et al.* (2020); Stergiadis *et al.* (2021). Vu sa simplicité et sa performance, nous utilisons cette technique pour passer, avec chaque phrase, le plongement de son domaine. À noter que ceci nécessite que le vecteur de méta-information soit de la même dimension que le plongement des mots.

Techniquement, et pour faciliter l'entrainement, nous avons plutôt créé 25 mots-étiquettes, pour les 25 domaines d'entrainement (voir la sélection des domaines d'entrainement à la section §4.4) et avons inséré au début de chaque phrase du corpus d'entrainement le mot-étiquette de son domaine. Avant l'entrainement, nous calculons le plongement des 25 domaines. Le modèle à entrainer est initialisé et son dictionnaire mots-plongements est mis à jour pour y rajouter les 25 mots-étiquettes et les plongements des domaines associés (le plongement pour chaque mot-étiquette est défini à celui calculé pour le domaine). Le modèle est ensuite entrainé. Pour le test, nous utilisons un 26^e mot-étiquette réservé pour le nouveau domaine.

Notre modèle présente divers avantages. Premièrement, il peut s'adapter à n'importe quel nouveau domaine. Deuxièmement, l'adaptation est possible même en présence d'un petit ensemble de phrases du domaine cible. Il s'adapte également très rapidement : aucun entrainement supplémentaire n'est requis et l'étape de calcul du plongement du domaine ne prend souvent que quelques secondes (la section 3.2.7 compare les vitesses des différentes

techniques). De plus, la méthode repose très peu sur la structure du modèle, et peut donc être facilement adaptable à d'autres architectures de type encodeur-décodeur. Finalement, notons que les résultats des expériences du chapitre 6 démontrent que le modèle ne souffre pas significativement d'oubli catastrophique.

Le modèle a par contre quelques limites. Premièrement, une phase de calcul de plongement doit être exécutée pour chaque nouveau domaine. Deuxièmement, la fonction de plongement est un module indépendant du modèle, qui n'utilise pas nécessairement les mêmes données. Également, un grand corpus de phrases parallèles est requis pour l'entrainement. Finalement, la technique ne fonctionne pas si la particularité du domaine se trouve principalement dans les phrases cibles et non sources, par exemple une tâche d'adaptation où l'on souhaite contrôler le niveau de politesse dans la traduction, comme dans Sennrich *et al.* (2016a).

3.2. Techniques de plongement

Différentes techniques de calcul de plongement de domaine existent, et nous en expérimentons cinq.

Chaque technique sera une fonction qui reçoit en entrée un multiensemble [1] de phrases, en langue *source* uniquement, issues du domaine, et génère en sortie son plongement sous forme de vecteur. Elle doit être suffisamment rapide pour pouvoir traiter plusieurs milliers de phrases en quelques secondes. Dans la logique de l'intuition des signaux présentée précédemment, il est attendu du plongement qu'il permette au modèle de reconnaitre les domaines similaires, de distinguer les domaines différents et de décortiquer les signaux qui le composent.

Le plongement est calculé *uniquement* à partir des phrases en langue source, puisque, selon notre problématique, seules celles-ci sont disponibles au moment du test. Nous ne croyons pas qu'il s'agit d'un obstacle au calcul d'une bonne représentation, car nous émettons l'hypothèse que le domaine peut généralement être déterminé entièrement, ou majoritairement, à partir de ses phrases sources. Cette hypothèse nous semble raisonnable puisque, dans la réalité, un traducteur professionnel peut reproduire dans sa traduction le style, le niveau de vocabulaire, les intentions, bref le domaine, uniquement en analysant les textes sources.

Le multiensemble de toutes les phrases sources d'un domaine est possiblement infini. Il n'est donc pas possible, contrairement à Macé et Servan (2019), d'utiliser l'ensemble du domaine pour calculer sa représentation. La technique de plongement doit donc pouvoir estimer au mieux le plongement du domaine à partir d'un sous-ensemble fini de phrases.

1. Un *multiensemble* est un ensemble où un même élément, une phrase dans notre cas, peut se retrouver plusieurs fois.

Dans les formules qui suivent, le domaine est représenté par d et le multiensemble des phrases issues du domaine est représenté par S_d. Le plongement estimé pour d est représenté par $\hat{D}_d$. La taille du plongement désiré est désignée par k.

Pour chaque technique, nous mettons entre parenthèses le nom sous lequel elle sera désignée dans le reste du mémoire.

Nous présentons ensuite à la section 3.2.7 les vitesses de calcul de chacune des techniques.

3.2.1. Moyenne des mots (μ-`mots`)

Cette technique, appelée également *SWEM-aver* (de l'anglais « *Simple Word Embedding Model – average* »), est proposée par Shen *et al.* (2018) et est celle utilisée dans Macé et Servan (2019).

Le plongement du domaine est calculé à partir de la moyenne des plongements de tous les mots des phrases de S_d. Soit M_d le multiensemble de tous ces mots. Soit également PLGMT(m) une fonction qui retourne le vecteur du plongement du mot m. Le plongement du domaine d est alors estimé par :

$$\hat{D}_d = \frac{1}{|M_d|} \sum_{m \in M_d} \text{PLGMT}(m) \tag{3.2.1}$$

Une fonction PLGMT est nécessaire pour obtenir le plongement des mots. Il existe déjà des modèles de plongement préentrainés que nous aurions pu utiliser, comme le modèle *word2vec* (Mikolov *et al.*, 2013) préentrainé par Google [2] ou le modèle *GloVe* (Pennington *et al.*, 2014) [3]. Macé et Servan (2019) affirment par contre obtenir de meilleurs résultats avec un modèle Transformer entrainé sur leurs données. Nous suivons leur exemple et entrainons premièrement un Transformer régulier sur tout le corpus d'entrainement [4]. Nous utilisons ensuite le dictionnaire mots-plongements appris par ce modèle comme fonction PLGMT.

Notons que nous utilisons la moyenne dans l'équation (3.2.1), mais que différentes fonctions d'agrégation seraient possibles. Par exemple, nous pourrions utiliser une agrégation globale de type MAX, ou une moyenne des agrégations MAX calculées sur chaque phrase (voir, par exemple, Shen *et al.*, 2018).

La technique a trois principaux avantages. Premièrement, le plongement est simple à calculer et ne nécessite pas de charger ou d'exécuter un modèle lourd. Une fois le modèle Transformer entrainé, seul le dictionnaire mots-plongements est nécessaire. Deuxièmement,

2. `https://code.google.com/archive/p/word2vec/`
3. `https://nlp.stanford.edu/projects/glove/`
4. Ce modèle sera en réalité le modèle `Base-BTC` présenté à la section §5.2

les plongements sont calculés à partir des mêmes données que celles utilisées par notre modèle. Finalement, Shen *et al.* (2018) démontrent que, malgré la simplicité de la technique, les plongements qu'elle calcule permettent souvent d'obtenir des performances égales ou supérieures à d'autres techniques de plongement plus complexes dans diverses tâches de traitement automatique de la langue naturelle (TALN).

Notons quand même quelques désavantages. Elle nécessite l'entrainement complet d'un modèle de TAN, et la qualité des plongements dépend de la qualité de l'ensemble d'entrainement. Elle ne tient pas compte de l'ordre des mots ni de leur rôle sémantique. En effet, l'équation (3.2.1) montre bien que l'ordre des mots n'a pas d'incidence sur le résultat. Différents mots ont différents sens selon leur utilisation, par exemple le mot « *love* » peut être un nom ou un verbe. Cette technique utilisera le même plongement dans ces deux cas. Finalement, l'agrégation de la moyenne donne le même poids à tous les mots. Les mots communs, mais qui offrent peu d'information sur le domaine, comme « *the* », « *a* » ou « *is* », peuvent bruiter les signaux caractéristiques du domaine.

Finalement, notons que si le modèle de TAN que nous entrainerons par la suite utilise des plongements de mots différents à ceux utilisés pour calculer le plongement du domaine, le lien entre les mots et le domaine sera perdu (Macé et Servan, 2019). Le dictionnaire mots-plongements du modèle de TAN que nous entrainerons devra donc être initialisé à celui de notre Transformer préentrainé, et nous le garderons figé durant l'entrainement.

3.2.2. Moyenne des mots contextualisés (μ-`mots-contexte`)

Cette deuxième technique, inspirée de μ-`mots`, utilise plutôt un plongement *contextuel* pour chaque mot. Un plongement contextuel est une représentation qui prend en compte la position et le rôle sémantique du mot dans la phrase. Par exemple, les plongements pour le mot « *love* » dans les phrases « *My love* » et « *I love* » pourraient être différents puisque le mot n'a pas le même rôle (un nom et un verbe).

La fonction qui calcule le plongement contextuel d'un mot, que nous appellerons $\text{PLGMT}^{\mathcal{C}}$, prend en paramètres le mot et la phrase où il se trouve. Pour chaque phrase $s \in S_d$, nous calculons une représentation $R(s)$ à partir de la moyenne des plongements contextuels de ses mots.

$$R(s) = \frac{1}{|s|} \sum_{m \in s} \text{PLGMT}^{\mathcal{C}}(m,s) \tag{3.2.2}$$

Le plongement du domaine est ensuite déterminé à partir de la moyenne de ces représentations de phrases.

$$\hat{D}_d = \frac{1}{|S_d|} \sum_{s \in S_d} R(s) \tag{3.2.3}$$

Des modèles de plongement contextuel préentrainés existent déjà, comme ELMo (Peters *et al.*, 2018), BERT (Devlin *et al.*, 2018) ou RoBERTa (Liu *et al.*, 2019). Nous souhaitons par contre utiliser notre corpus d'entrainement pour la création de la fonction PLGMT$^{\mathcal{C}}$. Un Transformer est premièrement entrainé sur ce corpus [5]. Ensuite, pour toute phrase s, nous la passons dans notre Transformer et nous définissons la valeur de PLGMT$^{\mathcal{C}}(m,s)$ comme étant le plongement calculé par l'*encodeur* pour le mot m. Ce plongement est effectivement contextuel grâce au module d'intra-attention de l'encodeur qui lui permet de considérer tous les mots de la phrase lors de la génération du plongement de chaque mot.

Notons là encore que dans l'équation (3.2.2), d'autres fonctions d'agrégation que la moyenne pourraient être utilisées, comme une agrégation de type MAX, par exemple.

Basé sur l'intuition exprimée, entre autres, dans Chinea-Rios *et al.* (2017) et Tars et Fishel (2018), nous utilisons la moyenne dans l'équation (3.2.3) car les vecteurs $R(s)$ calculés pour toutes les phrases d'un même domaine sont probablement regroupées en une grappe, et leur centroïde devient alors un bon représentant du domaine.

Les avantages de cette technique sont que la fonction de plongement est apprise à partir de nos données, et que les plongements sont contextuels.

Par contre, tout comme μ-`mots`, elle nécessite l'entrainement complet d'un modèle de TAN, et la qualité des plongements dépend de la qualité de l'ensemble d'entrainement. De plus, elle nécessite de charger en mémoire et d'exécuter un modèle Transformer pour calculer le plongement, un tel modèle pouvant nécessiter du matériel spécialisé. Notons par contre que seul l'encodeur a besoin d'être exécuté.

3.2.3. Sentence-BERT (`sbert`)

Cette technique utilise Sentence-BERT, un modèle basé sur BERT, pour calculer le plongement de chaque phrase, et nous utilisons leur moyenne comme plongement du domaine.

$$\hat{D}_d = \frac{1}{|S_d|} \sum_{s \in S_d} \text{SentenceBERT}(s) \tag{3.2.4}$$

BERT (Devlin *et al.*, 2018) est un modèle basé sur l'architecture Transformer et préentrainé sur un corpus unilingue. Le modèle est entrainé avec deux fonctions objectives. Pour la première, le décodeur doit retrouver un mot aléatoirement masqué dans la phrase d'entrée. Pour la deuxième, le modèle reçoit deux phrases (les deux phrases sont concaténées, mais

5. Ce modèle sera, encore une fois, le modèle `Base-BTC` présenté à la section §5.2

séparées par le mot spécial `[SEP]`) et il doit décider si elles se suivent ou non. Pour déterminer cela, l'encodeur est entrainé à produire pour le premier mot (le mot spécial `[CLS]` rajouté à chaque phrase) un plongement qui permet à un classificateur de déterminer si les phrases se suivent ou non. Devlin *et al.* (2018) démontrent ensuite qu'un tel modèle préentrainé peut ensuite être affiné pour diverses tâches de TALN.

Devlin *et al.* (2018) utilisent, entre autres, BERT pour calculer une similarité entre deux phrases. Ils affinent BERT sur une tâche supervisée de similarité en ajoutant un module de régression sur le plongement du mot spécial `[CLS]`. La sortie de ce module est un score de similarité. Les auteurs obtiennent de très bons résultats, mais cette technique n'est pas applicable à toutes les tâches de similarité. Si, par exemple, nous recherchons les deux phrases les plus similaires dans un ensemble de 10 000 phrases, le modèle devra être exécuté $n(n-1)/2 = 49\,995\,000$ fois – plus d'un an si le calcul prend une seconde par paire. D'autres chercheurs ont tenté d'utiliser le plongement de `[CLS]`, ou la moyenne des plongements des mots, comme représentation vectorielle de la phrase, pour ensuite utiliser des mesures de similarité vectorielle (voir, par exemple, May *et al.*, 2019; Qiao *et al.*, 2019; Zhang *et al.*, 2019). Mais Reimers et Gurevych (2019) démontrent que les résultats peuvent être pires que ceux obtenus avec une technique de plongement non contextuelle comme GLoVE (Pennington *et al.*, 2014).

Sentence-BERT (Reimers et Gurevych, 2019) offre une solution à ce problème. Il s'agit d'un modèle BERT affiné avec l'objectif de produire, pour une phrase d'entrée, une représentation vectorielle telle que deux phrases sémantiquement similaires auront des représentations similaires. Les vecteurs ainsi obtenus peuvent être comparés avec des mesures de similarités vectorielles pour déterminer la similarité des phrases.

Le modèle est entrainé sur deux tâches supervisées avec une architecture siamoise où deux phrases sont fournies aux modèles. La première tâche en est une d'inférence textuelle (« *Natural Language Inference* » en anglais), la deuxième est une tâche de similarité de phrase. Le modèle est premièrement entrainé sur la première tâche, puis sur la deuxième. Le plongement de la phrase est calculé par agrégation (la moyenne, par défaut) des sorties de l'encodeur.

L'avantage d'une telle méthode est que si, effectivement, les phrases d'un même domaine sont similaires, leur centroïde sera un bon représentant de leur domaine.

Un désavantage de cette technique est qu'un modèle BERT, une architecture lourde, doit être exécuté pour calculer le plongement. De plus, le modèle est entrainé sur un corpus externe, ce qui pourrait causer un biais si la distribution de nos données est trop différente de celles utilisées pour entrainer BERT et Sentence-BERT. Nous sommes également contraint par la langue de ce corpus : si nous utilisons un modèle Sentence-BERT entrainé en anglais, nous ne

pourrons pas l'utiliser dans une autre langue. Les auteurs ont par contre rendu disponibles des modèles Sentence-BERT préentrainés sur diverses langues [6].

Une autre difficulté est que nous ne contrôlons pas la taille du vecteur produit par Sentence-BERT. La version que nous utilisons produit un plongement de taille 768, alors que notre modèle nécessite un plongement de taille 512. Pour compresser le plongement produit par Sentence-BERT, une projection linéaire $L \in \mathbb{R}^{512\times768}$ lui est appliquée avant de l'insérer dans notre modèle. Cette projection est apprise conjointement avec les paramètres du modèle de traduction. Nous utilisons donc le plongement suivant :

$$\hat{D}_d^* = L\hat{D}_d \tag{3.2.5}$$

À noter que d'autres modèles préentrainés peuvent être utilisés pour obtenir un plongement de phrase, comme InferSent (Conneau *et al.*, 2017) ou Universal Sentence Encoder (Cer *et al.*, 2018).

3.2.4. Discriminateur de domaine (`pdd`)

Britz *et al.* (2017) et Zeng *et al.* (2018) amènent l'idée que deux informations peuvent être extraites de chaque phrase : une information spécifique à son domaine et une autre générique à tout domaine. Britz *et al.* (2017) proposent un modèle de traduction où un discriminateur de domaine est rajouté à l'encodeur afin de le forcer à inclure de l'information sur le domaine dans les plongements générés.

Inspirés de leur méthode, nous proposons une nouvelle technique de plongement au niveau de la phrase. Un modèle est entrainé pour générer, pour chaque phrase, un plongement qui inclut le plus d'information possible sur son domaine. Nous émettons l'hypothèse que le centroïde de ces vecteurs est une bonne représentation du domaine. Le plongement du domaine d est donc estimé par :

$$\hat{D}_d = \frac{1}{|S_d|}\sum_{s\in S_d} \mathrm{PDD}(s) \tag{3.2.6}$$

PDD (pour « Plongement Discriminateur de Domaine ») est simplement un encodeur de Transformer qui prend une phrase en entrée, et dont les plongements de sortie sont agrégés (par la moyenne) en un plongement représentatif de la phrase. Le modèle est entrainé à incorporer dans ce plongement le plus d'informations possible sur le domaine à l'aide d'un classificateur ajouté à sa sortie. Ce classificateur, entrainé conjointement au PDD, tente

6. `https://www.sbert.net/docs/pretrained_models.html`

de prédire le domaine d'origine de la phrase d'entrée. Le modèle complet est entrainé en maximisant la log-vraisemblance sur la classification.

Le classificateur est une projection linéaire $L \in \mathbb{R}^{n \times k}$, apprise conjointement avec le modèle, suivie d'un softmax pour obtenir une probabilité pour chacun des n domaines[7]. Nous avons opté pour une projection linéaire, car nous souhaitions que le module de classification reste simple afin de forcer le modèle à incorporer le plus d'information possible dans le plongement qu'il génère.

D'autres techniques que la moyenne sont possibles pour regrouper les plongements de la sortie de l'encodeur en un plongement de phrase. Il pourrait être intéressant, par exemple, de tester avec une agrégation de type MAX.

Les détails de l'entrainement du modèle sont rapportés à l'appendice E.

Les avantages de la technique sont qu'elle est entrainée sur les mêmes données que notre modèle final et qu'elle génère un plongement créé spécifiquement pour encoder l'information du domaine.

Les désavantages de la technique sont que la qualité du modèle est fonction de la taille et de la qualité du corpus d'entrainement, qu'elle nécessite d'entrainer un modèle, et qu'il faut exécuter un modèle Transformer pour obtenir une représentation.

3.2.5. TF-IDF (`tfidf`)

Comme dernière technique de calcul de plongement, nous souhaitions essayer une méthode plus traditionnelle. Elles sont parfois surprenantes !

Une idée simple est qu'un document, ou dans notre cas, un domaine, peut être représenté par les mots qu'il contient. Certains mots, par exemple « virus », se retrouveront plus dans un texte d'un domaine médical que dans un domaine artistique. Un domaine peut être représenté par un sac de mots où un compte est gardé pour chaque mot du vocabulaire. Nous pourrions naïvement croire que les mots les plus fréquents sont représentatifs du domaine, et que si ces mots sont également communs dans un autre domaine, alors ils sont semblables. Nous ferions erreur, car les mots les plus répétés dans un corpus sont souvent les moins significatifs. Par exemple, le mot « *the* » sera probablement le mot le plus populaire de chaque domaine.

Karen Sparck Jones propose en 1972 (Jones, 1972) une mesure statistique au niveau du mot dont l'intuition est que si un mot se retrouve dans plusieurs documents, il n'est pas une bonne information sur celui-ci, et son poids devrait être plus faible. La mesure, qui sera appelée IDF (de l'anglais « *Inverse Document Frequency* ») sera éventuellement combinée

7. Pour le `ppd` que nous avons entrainé, $n = 25$ et $k = 512$.

avec la mesure TF (de l'anglais « *Term Frequency* »), qui, elle, augmente avec le nombre de fois que le terme apparait dans un document.

Soit un ensemble de domaines D. Pour un domaine $d \in D$, la mesure TF-IDF du mot m dans ce domaine est :

$$\text{TF-IDF}(m,d,D) = \frac{|\{m_i : m_i = m, m_i \in d\}|}{|d|} \log \frac{|D|}{|\{d_j : m \in d_j, d_j \in D\}|}, \tag{3.2.7}$$

c'est-à-dire [la proportion des mots de d qui sont m], multiplié par le logarithme de [le nombre de domaines] divisé par [le nombre de domaines où le mot m apparait]. Cette mesure peut être utilisée comme compte dans une représentation sac de mot. Nous appellerons une telle représentation une « représentation TF-IDF ». Celle-ci peut être vue comme un vecteur ou chaque élément correspond à un mot du vocabulaire.

Cette technique est très populaire en recherche d'information (Beel *et al.*, 2016). Un score entre deux documents est alors utilisé pour déterminer leur similarité (Schütze *et al.*, 2008b). Ce score peut être la distance cosinus entre leur représentation TF-IDF. Cette distance est préférée à la distance euclidienne puisqu'un document avec plus de mots aura une norme vectorielle plus grande qu'un petit document, même s'ils sont similaires.

Notre dernière technique de plongement sera donc la représentation TF-IDF.

Par contre, la taille du vecteur TF-IDF est égale au nombre de mots dans le vocabulaire, qui peut atteindre plusieurs dizaines de milliers d'éléments. Notre modèle nécessite un plongement de 512 éléments. Une réduction du vecteur est possible avec le procédé d'analyse sémantique latente (LSA, de l'anglais « *Latent Semantic Analysis* ») (Schütze *et al.*, 2008a). Une matrice de type mot×domaines peut être réduite en une matrice concepts×domaines dont le nombre de « concepts » peut être déterminé (512, dans notre cas).

L'appendice F présente les détails d'implémentation de la technique.

Les avantages de cette technique sont que le plongement est facile à calculer et que la méthode TF-IDF est un algorithme mature, avec une littérature riche et un large support parmi les outils de développement.

Parmi les désavantages, notons que nous avons besoin d'un corpus d'entrainement et d'un processus de prétraitement des textes spécifique. Également, une représentation TF-IDF est un sac de mots où l'ordre et le rôle sémantique des mots sont perdus lorsque l'on utilise une segmentation unigramme.

3.2.6. Considération technique

La technique μ-`mots` (3.2.1) requiert que le modèle de traduction qui sera entrainé avec ses plongements utilise le même dictionnaire mots-plongements que celui qu'elle a utilisé. Le dictionnaire devra, de plus, être figé durant l'entrainement.

Pour comparer équitablement les différentes techniques, la même stratégie sera utilisée pour chaque modèle de traduction entrainé sur chacune des techniques de plongement. En d'autres mots, pour chaque modèle de traduction à entrainer, le dictionnaire mot-plongements sera initialisé à celui appris par le modèle de comparaison `Base-BTC` (voir section §5.2) et ses paramètres resteront figés durant l'entrainement.

3.2.7. Vitesses de calcul

	CPU (sec)	GPU (sec)
μ-`mots`	0,091	n/a
μ-`mots-contexte`	2,241	0,239
`sbert`	63,198	2,529
`pdd`	2,272	0,159
`tfidf`	1,563	n/a

Tableau 3.1. Vitesse de calcule (en secondes) du plongement pour chaque technique sur un ensemble de 1 000 phrases. Colonne « CPU » : temps de calcul sur un processeur régulier. Colonne « GPU » temps de calcul sur un processeur graphique est utilisé (NVIDIA TITAN V). Applicable seulement pour certaines techniques.

Chapitre 4

Les données

Nous présentons dans ce chapitre les données avec lesquelles nous avons travaillé, comment nous les avons nettoyées et prétraitées. De plus, puisque notre objectif est de créer un modèle capable de s'adapter à tout nouveau domaine, nous avons besoin de tester le modèle sur des domaines exclus de l'entrainement. Nous expliquons comment nous les avons choisis.

Nous avons eu la chance de travailler avec des données prêtées généreusement par le Bureau de la traduction du Canada [1] (BTC). Le BTC est une institution fédérale qui a pour mission « *de servir les ministères et autres organismes institués par une loi fédérale ou par un décret en conseil, ainsi que les deux Chambres du Parlement, pour tout ce qui concerne la traduction et la révision de leurs documents* » (Gouvernement du Canada, 2018).

Travailler avec leurs données est une chance puisque les recherches en traduction automatique se font souvent avec les mêmes corpus d'entrainement, extraits de sources libres comme Wikipédia (Wołk et Marasek, 2014) ou le projet OpenSubtitles (Tiedemann, 2012). Ces corpus ne sont pas représentatifs de ceux avec lesquels l'industrie de la traduction travaille. Par exemple, la figure 4.1 montre la différence entre la distribution de la longueur des phrases dans le corpus de Wikipédia et la distribution dans le corpus du BTC (décrit plus bas). Un modèle entrainé sur un de ces corpus sera biaisé si utilisé sur l'autre. En travaillant avec les données du BTC, nous pourrons rendre compte des performances de notre modèle dans un contexte réaliste pour les organismes de traduction. De plus, le corpus qui nous a été fourni est un ensemble déjà aligné de phrases traduites par des professionnel·le·s de la traduction.

La grande majorité des données du BTC sont des phrases en anglais traduites vers le français. De plus, les modèles préentrainés que nous utiliserons parfois sont souvent entrainés uniquement sur des corpus anglais, ou dans la direction anglais→français. Nous nous concentrerons donc sur un modèle qui traduit de l'anglais vers le français.

1. `https://www.tpsgc-pwgsc.gc.ca/bt-tb/index-fra.html`

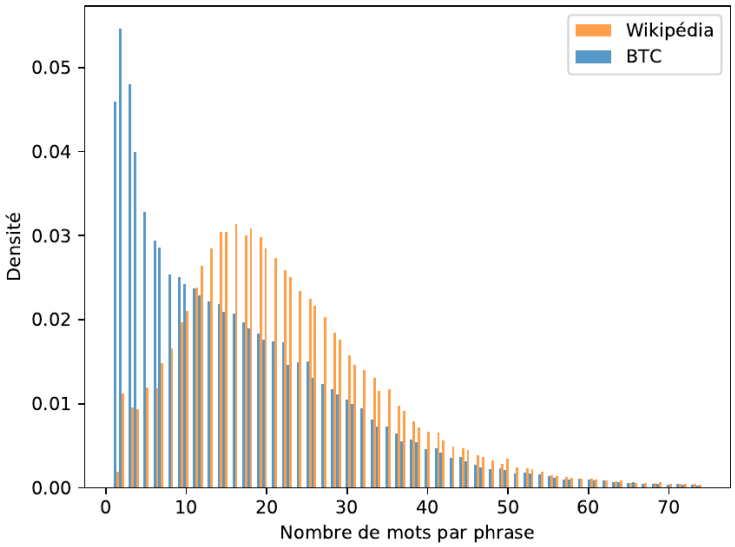

Figure 4.1. Distribution de la longueur des phrases anglaises selon le corpus du BTC et celui de Wikipédia. Les phrases sont segmentées (« *tokenized* » en anglais) à l'aide de Moses (Koehn *et al.*, 2007).

4.1. Origine des données et leur domaine

Le BTC nous a donné accès à deux bases de données : une liste de commandes de traduction et une *mémoire de traduction*.

La liste des commandes est une sélection partielle des projets de traduction complétés par le BTC, sur la période d'avril 2014 à février 2019. Chaque commande est le détail technique d'une demande d'un client et contient plusieurs tâches, dont des tâches de traduction. À chaque tâche sont assignées une ou plusieurs catégories, dites « spécialités », tels « Militaire », « Emploi », « Médecine », « Juridique », etc. Nous avons extrait 32 spécialités différentes et nous les listons à l'appendice A.

La mémoire de traduction, quant à elle, se présente sous la forme d'un ensemble de plus de 1,8 million de fichiers TMX. Une mémoire de traduction est une base de données de segments de textes dans une langue source et leurs traductions associées dans une ou plusieurs langues cibles. Une telle mémoire peut ensuite être utilisée pour faciliter et accélérer la traduction de futurs textes. Un fichier TMX, quant à lui, est un document XML qui liste des phrases en langue source et, pour chacune, sa traduction en une ou plusieurs langues cibles. Plus de 82 % des fichiers TMX du BTC sont des traductions de l'anglais vers le français. Ces fichiers contiennent au total plus de 115 millions de paires de phrases. Notons qu'en général, il n'y a aucun ordre logique entre les phrases et qu'elles ne se suivent pas.

Chaque fichier TMX est associé à une tâche, et donc par extension, chaque phrase peut être associée à une ou plusieurs spécialités.

Le concept de « spécialité » utilisé par le BTC semble correspondre à notre concept de domaine. En effet, les différentes spécialités constituent des corpus qui varient en sujet. Nous avons donc décidé de regrouper en « domaines » les phrases qui étaient assignées à la même spécialité.

Par contre, nous n'avons pas gardé toutes les phrases de la mémoire de traduction. Premièrement, plusieurs fichiers TMX sont associés à une tâche que nous n'avions pas dans notre liste de commandes. Deuxièmement, nous avons ignoré la spécialité « TAG - Textes administratifs et généraux ». Son nom laisse présager qu'elle est composée de sujets disparates, et elle est, de loin, celle avec le plus de tâches. Pour ces raisons, nous l'avons considéré comme une spécialité « fourre-tout » et l'avons éliminée. Finalement, nous avons éliminé les phrases qui étaient associées à plusieurs spécialités, afin d'éviter de bruiter inutilement notre ensemble de données.

Ce premier ensemble de phrases parallèles est composé de 31 504 841 de paires de phrases anglais→français réparties en 31 domaines.

4.2. Filtrage des données

Une partie importante du corpus contient des paires invalides ou corrompues. Ces erreurs se sont probablement insérées lors de la traduction ou lors de l'exportation de la mémoire de traduction.

Près du tiers des paires n'ont pas de traduction française (10 571 440 paires). Les détecter est simple et nous les éliminons. Un petit nombre (environ une centaine) n'ont pas de phrase source anglaise, et elles sont également éliminées.

Dans l'ensemble restant, nous remarquons que certaines paires semblent mal alignées (la traduction ne correspond pas à la phrase source), que pour certaines la phrase source n'est pas traduite (la phrase cible française est la même que la phrase source), et que d'autres sont corrompues – une de leurs phrases contient des symboles incompréhensibles.

Pour déceler les phrases problématiques, nous utilisons un modèle préentrainé proposé par Grégoire et Langlais (2018). Le modèle calcule, à partir d'une phrase et de sa traduction candidate, un score d'équivalence de traduction. Si celui-ci est inférieur à un seuil, nous considérons que la traduction est invalide et la paire est éliminée. Le modèle est un BiRNN (Schuster et Paliwal, 1997) siamois entrainé pour produire une représentation sémantique de chaque phrase et calculer un score d'équivalence à partir de celles-ci. De façon heuristique, nous avons établi le seuil à 0,1. Nous avons ainsi éliminé 1 086 104 paires de phrases.

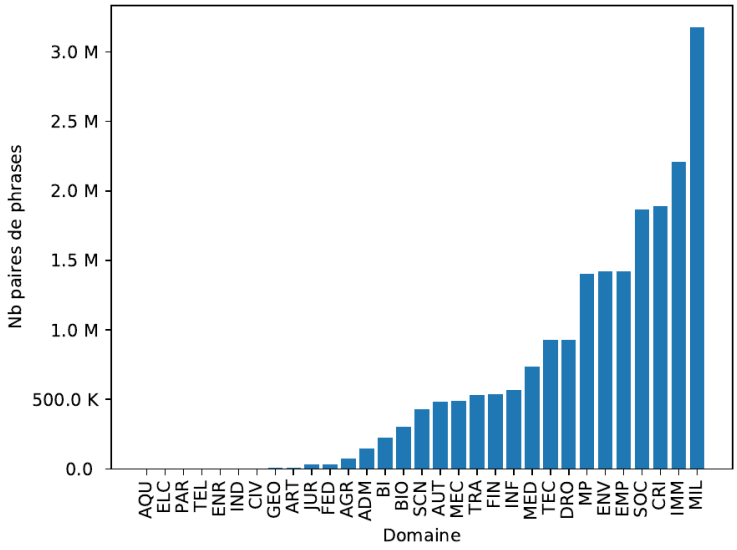

Figure 4.2. Distribution des données filtrées parmi les domaines.

Après ce filtrage, nous obtenons un corpus de 19 847 297 paires de phrases. L'appendice B détaille la distribution des phrases parmi les domaines et la figure 4.2 montre graphiquement cette distribution.

4.3. Prétraitement des données

La qualité des hypothèses produites par notre modèle de traduction dépend directement de la qualité de notre corpus d'entrainement. Nous avons vu au chapitre 1 que les modèles de TAN ont besoin de grandes quantités de données pour apprendre les relations complexes entre la phrase d'entrée et celle de sortie. Plus un corpus est bruité, plus le modèle aura de la difficulté à découvrir ces relations pour un corpus de taille donnée.

Le bruit dans un ensemble de textes peut se présenter sous différentes formes. Certains symboles peuvent être représentés par différents caractères. Par exemple, le guillemet double peut être représenté par ", «, », “, ”, etc. Certains mots n'apportent parfois aucune information utile et peuvent complexifier la tâche de traduction. Par exemple, les adresses web et courriel n'ont généralement pas à être traduites, mais leur présence peut créer des informations contradictoires pour le modèle – si une adresse web contient des mots anglais, par exemple. Des erreurs typographiques peuvent également se glisser dans les phrases, comme un guillemet superflu en fin de phrase.

Ces erreurs et certaines autres ont été corrigées dans le corpus à l'aide d'un script de nettoyage que nous avons développé et qui est détaillé à l'appendice D.

4.4. Sélection des domaines de test et de validation

Nous avons pour objectif d'entrainer un modèle de traduction qui peut ensuite s'adapter à tout nouveau domaine, même si celui-ci n'a pas été vu durant l'entrainement. Pour tester un tel modèle sur nos données, nous avons besoin de garder à l'écart un certain nombre de domaines qui ne seront pas utilisés pour l'entrainement, et que nous appellerons « les domaines de test ». Pour nous permettre de sélectionner les hyperparamètres de notre modèle, nous avons également besoin d'un certain nombre de « domaines de validation ».

Dans l'objectif d'avoir des domaines variés pour le test, mais sans trop réduire l'ensemble d'entrainement, nous avons choisi, parmi les 31 domaines, 3 domaines de test et 3 domaines de validation. Les 25 autres domaines sont gardés pour l'entrainement. Dans le chapitre 6, nous testerons également notre modèle sur des domaines extérieurs aux données du BTC (voir section 6.4.3).

Avant de détailler comment nous avons sélectionné les domaines, nous souhaitons souligner que certains auteurs (voir, par exemple, Tars et Fishel, 2018; Aharoni et Goldberg, 2020) critiquent la division des données à partir d'une structure de domaines établie par des humains. Cette structure, bien que pratique, peut-être mal faite, peut contenir beaucoup de bruit, dû à des erreurs humaines ou à une interprétation différente entre les traducteurs, et peut ne pas être adaptée à une tâche d'apprentissage automatique. Ces auteurs proposent plutôt de rediviser, de manière non supervisée, le corpus en pseudodomaines, uniquement à partir de l'information de la phrase. Tars et Fishel (2018) utilisent *sent2vec* (Pagliardini *et al.*, 2018) pour calculer un vecteur pour chaque phrase et divisent ensuite les données en k domaines à l'aide d'un partitionnement en k-moyennes. Aharoni et Goldberg (2020) génèrent une représentation vectorielle de chaque phrase à l'aide de BERT (Devlin *et al.*, 2018) ou RoBERTa (Liu *et al.*, 2019) et regroupent les vecteurs en k domaines à l'aide d'un mélange de gaussiennes (« *Gaussian mixture model* »). Malgré l'intuition logique de ces approches, nous sommes restés avec la structure de domaines définie ci-dessus afin de garder l'aspect intuitif de la catégorisation actuelle, et aussi pour des raisons de temps et de ressources.

Nous nous sommes par contre inspirés de leur idée pour sélectionner nos domaines. Ces auteurs définissent un domaine à partir de la distribution de ses phrases (en langue source, dans notre cas). En modélisant cette distribution pour chaque domaine, nous pouvons utiliser la similarité entre les distributions comme substitut à la similarité entre les domaines. Les domaines de test peuvent ensuite être sélectionnés selon leur degré de similarité aux autres domaines et selon nos intérêts.

Une mesure populaire de la différence entre deux distributions est la divergence de Kullback-Leibler (« divergence KL »). Dans le cas continu sur $\mathbb{R}^n$, la divergence KL d'une distribution P par rapport à une distribution Q est donnée par :

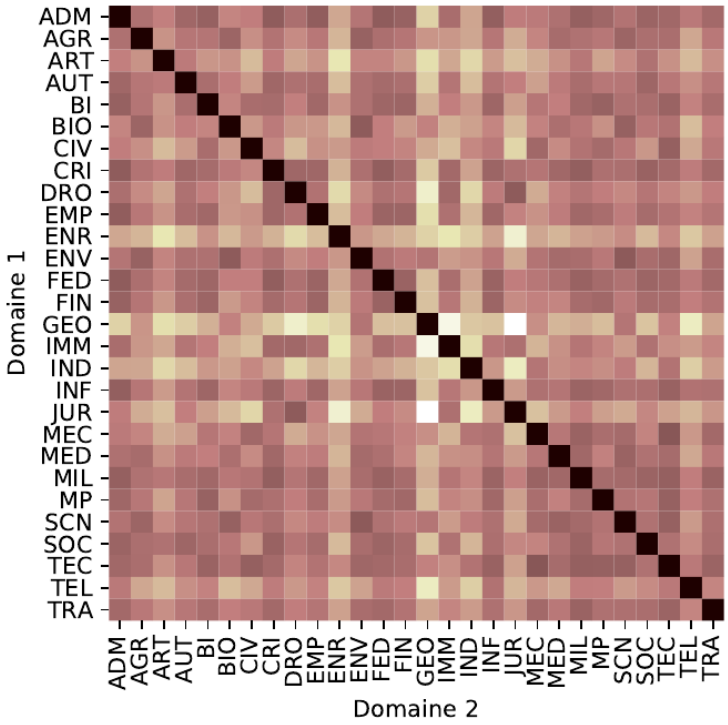

Figure 4.3. Similarité entre les domaines basée sur la divergence KL. Plus la case est foncée, plus les domaines sont similaires. Les 3 domaines avec moins de 1 000 paires de phrases ne sont pas affichés.

$$D_{KL}\left(P \parallel Q\right) = \int_{\mathbb{R}^n} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x} \tag{4.4.1}$$

Plus les distributions sont semblables, plus la valeur de la divergence est basse. Elle vaut 0 si les distributions sont identiques. L'appendice G détaille comment nous avons calculé la divergence KL entre chaque domaine.

La divergence KL n'est pas symétrique, dans le cas général, $D_{KL}\left(P \parallel Q\right) \neq D_{KL}\left(Q \parallel P\right)$. Pour faire une mesure de similarité symétrique entre les domaines, nous utilisons (la négation de) la moyenne de leurs deux divergences KL. Par exemple, la similarité entre les domaines d_1 et d_2 (avec distributions de phrases P_1 et P_2 respectivement) est :

$$\text{sim}(d_1,d_2) = -\frac{1}{2}\left[D_{KL}(P_1 \parallel P_2) + D_{KL}(P_2 \parallel P_1)\right] \tag{4.4.2}$$

(Nous inversons le signe de la moyenne afin d'obtenir une valeur qui augmente avec la similarité, et non qui diminue.)

La figure 4.3 présente graphiquement la similarité calculée entre chaque domaine.

Notons que d'autres mesures de similarité existent et l'on pourrait se demander si elles auraient donné des résultats différents. Nous effectuons une expérience à la section §5.3 qui démontre que nous obtenons des résultats semblables avec d'autres mesures de similarité. Notons également que nous n'avons pas utilisé les techniques de plongement présentées à la section §3.2 pour déterminer les similarités. Ces techniques sont des compressions et elles gardent très peu d'information sur la distribution des phrases au sein du domaine.

À partir des similarités calculées, nous choisissons 3 domaines de test de 3 domaines de validation. Pour éviter des tests trop volatiles, nous avons exclu des candidats les 3 domaines qui ont moins de 1 000 paires (AQU, ELC, PAR). Nous avons aussi tenté d'éviter de choisir les plus gros domaines afin de garder le plus gros corpus d'entrainement possible.

Pour les domaines de test, nous choisissons :

MEC: Très proche du domaine TEC, MEC va nous permettre de tester les performances du modèle sur un domaine très semblable à un utilisé pour l'entrainement du modèle.

GEO: La moyenne de sa similarité avec tous les autres domaines montre qu'il est globalement « éloigné » des domaines utilisés pour l'entrainement. GEO nous permettra de tester le modèle sur un domaine peu représenté dans le corpus d'entrainement.

FED: En opposition à GEO, ce domaine est globalement « proche » des domaines d'entrainement. FED nous permettra de tester le modèle sur un domaine typique à l'ensemble d'entrainement.

Pour les domaines de validation, nos critères sont assez semblables :

ENR: Domaine globalement dissimilaire aux domaines d'entrainement.

ADM: Domaine globalement similaire aux domaines d'entrainement.

CIV: Domaine à similitude moyenne aux domaines d'entrainement.

4.5. Division en ensembles d'entrainement de validation et de test

En plus des performances sur les *domaines de test*, nous souhaitons évaluer les performances du modèle sur les *domaines d'entrainement*, afin de vérifier la présence de l'oubli catastrophique.

Nous devons faire bien attention de ne pas évaluer les performances du modèle uniquement sur les données d'entrainement. L'erreur sur cet ensemble permet l'entrainement du modèle, car l'algorithme cherche la paramétrisation permettant de la réduire. Mais l'objectif de l'apprentissage n'est pas d'optimiser cette erreur, mais de s'assurer que le modèle pourra bien généraliser sur de nouvelles données (Bengio, 2012b). Pour estimer cette généralisation, le

modèle est testé, après entrainement, sur un ensemble appelé « test ». Il est important que les données de l'ensemble de test soient exclues de l'ensemble d'entrainement pour éviter de biaiser l'estimation de l'erreur de généralisation. De plus, pour aider à sélectionner les meilleurs hyperparamètres, le modèle est testé sur un ensemble de données appelé « validation », qui doit être distinct de ceux d'entrainement et de test.

Nous devons donc créer, pour chaque domaine d'entrainement, un ensemble de test et de validation.

Pour ces domaines, 2 000 de leurs paires, ou 10 % selon le plus petit nombre, sont sélectionnées aléatoirement pour créer un ensemble de test. Nous choisissons ensuite 0,5 % de leur corpus pour créer un ensemble de validation. Le reste est gardé pour l'entrainement.

Avec une telle division, la distribution des domaines au sein de l'ensemble d'entrainement est différente de celle de l'ensemble formé de toutes les données de test (ou de validation). Ceci serait un problème si l'on testait le modèle sur cette union des ensembles de test, car notre erreur serait biaisée. Nous n'aurons pas ce problème, car, au chapitre 6, nous nous limitons à tester le modèle sur chaque domaine individuellement, et jamais sur l'union de tous les ensembles. Notre distribution des données de validation, quant à elle, respecte la distribution des données initiales, car nous utiliserons l'union de toutes les données pour obtenir un score unique de validation.

La figure 4.2 nous a précédemment montré la répartition des données. Nous remarquons un clair déséquilibre entre les domaines. Le plus gros domaine contenant plus de trois millions de paires, contre 274 pour le plus petit. Ce déséquilibre se retrouve aussi dans les données d'entrainement. Rééquilibrer le corpus permet de limiter le biais favorisant les gros domaines et d'augmenter les performances sur les plus petits. Différentes méthodes de rééquilibrage existent. Par exemple, certaines techniques présentées à la section §2.2 peuvent être utilisées pour rechercher et transférer des données des plus gros domaines vers les plus petits.

Une solution plus simple, mais efficace, consiste à réduire les plus gros domaines en éliminant des données, et à augmenter artificiellement les plus petits en dupliquant certaines de leurs données. Arivazhagan *et al.* (2019) décrivent une stratégie dite de « température » pour recréer un nouveau corpus plus équilibré. Un domaine est aléatoirement choisi, puis une de ses données est aléatoirement sélectionnée et ajoutée au nouveau corpus. Le processus est repris jusqu'à obtenir un corpus de la taille souhaitée. Mais au lieu de définir la probabilité de choisir un domaine à partir de sa taille relative, c'est-à-dire $p(d_i) = \frac{|d_i|}{\left|\bigcup_j d_j\right|}$, nous utilisons une probabilité proportionnelle à $p(d_i)^{\frac{1}{T}}$ où T est un paramètre contrôlant le degré d'uniformité entre les domaines. Une valeur de $T = 1$ équivaut à la distribution initiale, mais une grande valeur, comme $T = 100$, produit une distribution quasi uniforme. La figure 4.4 montre l'effet de diverses valeurs de T sur la distribution de nos données.

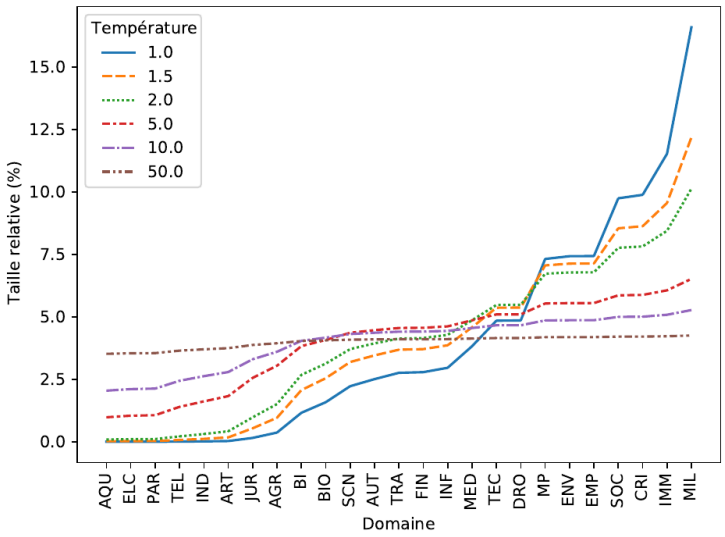

Figure 4.4. Effet sur la taille relative des domaines selon différentes valeurs de température. Les domaines sont triés selon leur taille originale.

Nous utilisons cette stratégie pour rééquilibrer nos données et choisissons, comme dans Lample et Conneau (2019), une valeur $T = 2$. La figure 4.5 montre la distribution des données obtenues. Il y a encore un clair déséquilibre, mais nous ne voulions par utiliser une valeur de T trop grande. Nous avions peur de trop dupliquer les plus petits domaines et de créer une situation où le modèle voit toujours les mêmes phrases associées aux mêmes domaines, et qu'il tombe en surapprentissage. Nous aurions également pu enlever les plus petits domaines, mais cela aurait réduit notre variété de domaines d'entrainement.

Pour ce qui est des ensembles des domaines de test, nous fixons leur taille à 5 700 paires, choisies aléatoirement, afin de nous assurer de comparer équitablement les performances du modèle sur chacun d'eux. Pour les domaines de validation, nous fixons leur taille à 1 600 paires. Ces valeurs ont été choisies selon le plus petit domaine de chaque groupe.

L'appendice C présente la distribution finale de nos données.

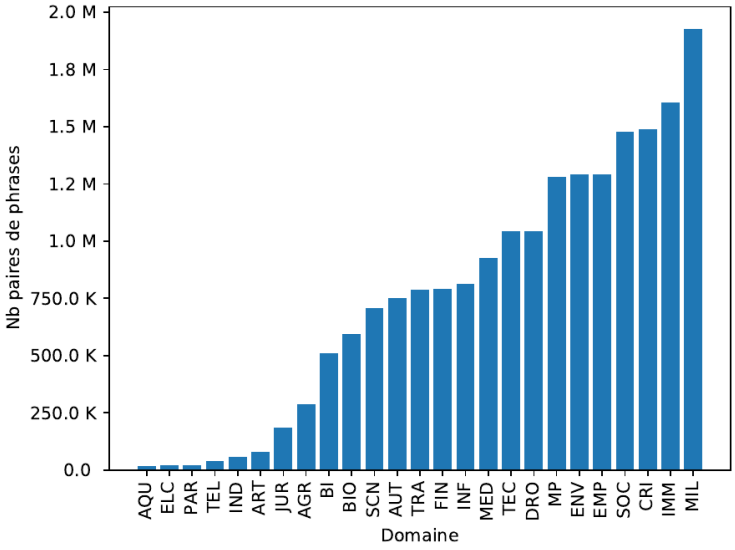

Figure 4.5. Distribution des données d'entrainement (des domaines d'entrainement) après rééquilibrage par température ($T = 2$). À comparer avec la distribution sans rééquilibrage de la figure 4.2.

Chapitre 5

Les modèles de comparaison (*baselines*)

Nous présentons dans ce chapitre les modèles de comparaison (« *baseline models* » en anglais) qui nous permettront de mieux évaluer les gains obtenus ou les pertes subies par l'utilisation de notre modèle.

Les modèles décrits ici ont été choisis dans l'objectif de simuler diverses stratégies qui pourraient être utilisées par l'industrie pour la traduction d'un nouveau domaine. Ils ont été sélectionnés selon deux principaux critères : ils devaient pouvoir s'adapter (s'il y a lieu) au nouveau domaine de façon non supervisée (uniquement avec des données en langue source), et ils ne devaient pas nécessiter d'entrainement additionnel à chaque nouveau domaine.

5.1. Modèle préentrainé sur un corpus extérieur (`Base-WMT`)

Selon le *2020 European Language Industry survey* (ELIA *et al.*, 2020), une enquête menée auprès de 809 acteurs du milieu de la traduction, 77 % des départements de traduction affirment utiliser uniquement ou majoritairement des outils externes dans leur travail. Des systèmes de traduction automatique tiers, comme DeepL [1], sont nommés comme outils pour générer des traductions qui seront ensuite travaillées dans une étape de postédition. De tels produits ne sont souvent pas adaptables et doivent être utilisés tels quels.

Notre premier modèle de comparaison simule l'utilisation d'un tel système externe de traduction. Il s'agit du modèle de TAN anglais→français préentrainé par Ott *et al.* (2018) sur le corpus WMT14 (anglais→français) [2]. Le modèle est un Transformer (Vaswani *et al.*, 2017) implémenté dans la librairie fairseq (Ott *et al.*, 2019). La structure du modèle (sa « taille »)

1. `https://www.deepl.com/translator`
2. `https://www.statmt.org/wmt14/translation-task.html`

correspond à la variation « *big* » décrite dans Vaswani *et al.* (2017). En particulier, les plongements internes et ceux des mots sont de taille 1024 et le module d'attention utilise 16 têtes. Le modèle possède près de 222 millions de paramètres.

Plutôt que d'entrainer le modèle comme dans Vaswani *et al.* (2017), les auteurs utilisent diverses stratégies, comme les calculs en nombre flottant à demi-précision, et l'accumulation de gradients sur plusieurs mini-lots, pour accélérer jusqu'à 5 fois le temps d'entrainement du Transformer.

L'entrainement se fait sur le corpus WMT14 anglais→français, un ensemble de plus de 36 millions de paires de phrases parallèles. Un vocabulaire de 40 000 mots est construit par factorisation BPE (Sennrich *et al.*, 2016b). Cette technique construit un vocabulaire à partir des *segments de mots* les plus fréquents du corpus. Ceci permet de représenter des mots absents du corpus d'entrainement comme une combinaison de segments connus. Le modèle est entrainé pendant 29 époques pour un peu plus de 62 000 itérations.

Nous avons choisi ce modèle, car, à sa publication, il atteignit un nouveau record sur la tâche de traduction WMT14 anglais→français, il est accessible publiquement [3], et nous avons accès à ses paramètres et son architecture, ce qui nous permettra de l'affiner (voir la section §5.3).

5.2. Modèle préentrainé sur le corpus du BTC (`Base-BTC`)

Comme expliqué au chapitre 4, les ensembles d'entrainement utilisés par les modèles de traduction prêts à l'emploi, comme le `Base-WMT`, ne sont souvent pas représentatifs des textes que des organisations comme le BTC ont à traduire. Les traductions produites par de tels modèles peuvent donc être biaisées.

Notre deuxième comparaison est donc un modèle de TAN entrainé uniquement sur les données du BTC. Tout le corpus d'entrainement des domaines d'entrainement est utilisé, mais sans porter aucune attention au domaine de chaque phrase. Nous avons la chance d'avoir accès à un ensemble de plus de 19 millions de paires, ce qui nous permet d'entrainer un modèle complet sans avoir recours à des stratégies d'augmentation de corpus.

Le modèle est un Transformer implémenté dans la librairie fairseq (Ott *et al.*, 2019). Nous utilisons deux optimisations décrites dans Ott *et al.* (2018) — calcul en demi-précision et accumulation sur 8 lots — et entrainons le modèle sur un seul processeur graphique. La structure du modèle (sa « taille ») correspond à la variation « *base* » décrite dans Vaswani *et al.* (2017) pour un total d'un peu plus de 64 millions de paramètres. Ce modèle est donc « plus petit » que celui de `Base-WMT`, mais notre ensemble d'entrainement est aussi plus

3. `https://github.com/pytorch/fairseq/blob/master/examples/scaling_nmt/README.md`

restreint ; un modèle avec plus de capacité aurait pu tomber en surapprentissage. De plus, vu la limite du matériel à notre disposition, entrainer un plus gros modèle aurait pris un temps trop important. Les résultats exposés au chapitre 6 démontrent de toute façon que ce modèle est compétitif à `Base-WMT`.

Le modèle est entrainé (sur l'ensemble d'entrainement des domaines d'entrainement, voir section §4.5) sur un seul processeur graphique (TITAN V) jusqu'à ce que l'erreur de validation ne s'améliore pas pour au moins 1,5 époque d'entrainement. Le modèle est entrainé pour 18 époques (204 000 itérations) en environ 48 heures. L'appendice I détaille la configuration de ce modèle. Un vocabulaire de 40 000 mots est construit par factorisation BPE (Sennrich *et al.*, 2016b).

Au moment du test, nous utilisons un modèle construit à partir de la moyenne des 10 derniers points de contrôle (« *checkpoint* » en anglais). Un point de contrôle est créé à toutes les 1 000 itérations.

5.3. Modèle affiné sur le domaine le plus proche (`Base-WMT-[XXX]` et `Base-BTC-[XXX]`)

L'affinage – méthode que nous avons vue au chapitre 2 – est l'une des techniques les plus simples, mais aussi les plus efficaces, en adaptation de domaine. Un modèle de traduction préentrainé est affiné sur les données du nouveau domaine pour obtenir le modèle spécialisé.

Nous ne pouvons par contre pas appliquer cette technique directement à notre situation. Premièrement, l'affinage est une méthode de transfert d'apprentissage supervisé ; elle nécessite un corpus parallèle dans le domaine cible. Nous n'avons qu'un ensemble de textes unilingues. Deuxièmement, elle nécessite l'entrainement d'un nouveau modèle pour chaque nouveau domaine, ce qui est couteux en temps et en ressources, ce que nous tentons d'éviter.

Nous proposons plutôt d'affiner un modèle préentrainé différent sur chaque *domaine d'entrainement*. Donc le modèle préentrainé est dupliqué 25 fois, et chaque copie est affinée sur un des 25 domaines d'entrainement (en réalité, nous avons affiné seulement sur les domaines d'intérêt). Au moment de traduire un nouveau domaine, nous recherchons parmi les domaines d'entrainement le plus similaire et utilisons son modèle affiné pour traduire les textes. Cette stratégie requiert l'entrainement de plusieurs modèles, mais, une fois prêts, aucun nouvel entrainement n'est nécessaire.

Diverses techniques existent permettant de trouver le domaine le plus similaire. Nous avons proposé, à la section §4.4, une procédure basée sur la divergence KL. Nous reprenons ici la même technique pour trouver les domaines les plus proches pour chaque domaine de test. Les résultats sont présentés au tableau 5.1.

	1er domaine le plus similaire	2e domaine le plus similaire
MEC	**TEC**	TRA
FED	**CRI**	FIN
GEO	**ENV**	SCN

Tableau 5.1. Domaines d'entrainement les plus similaires pour chaque domaine de test, selon la similarité basée sur la divergence de KL présentée à la section §4.4.

Est-ce que la divergence KL permet de bien mesurer la similarité entre les domaines ? Est-ce qu'une autre mesure de similarité aurait donné des résultats différents ? Nous présentons à l'appendice H une courte étude que nous avons réalisée et qui compare différentes mesures de similarités. Les résultats montrent que toutes les techniques sont en accord avec le premier ou le deuxième choix de la divergence KL. Nous concluons qu'il s'agit d'une mesure de similarité crédible.

Pour chacun des domaines de test, nous initialisons un nouveau modèle à partir des paramètres du modèle préentrainé `Base-WMT`. Nous poursuivons son entrainement (affinage) sur les données du 1er domaine le plus similaire jusqu'à ce que l'erreur de validation ne s'améliore pas pour au moins 10 époques d'entrainement. Nous nommons ces modèles `Base-WMT-TEC`, `Base-WMT-CRI` et `Base-WMT-ENV`.

À des fins de comparaison, nous effectuons le même processus avec le modèle préentrainé `Base-BTC`. Nous nommons ces modèles `Base-BTC-TEC`, `Base-BTC-CRI` et `Base-BTC-ENV`.

Les détails de la configuration de ces 6 modèles sont présentés à l'appendice I.

Chapitre 6

Expériences, évaluations et analyses

Dans ce chapitre, nous évaluons les performances des modèles de comparaison et celles du modèle que nous proposons. Nous les comparons et analysons en détail les résultats obtenus.

Mais en premier lieu, pour évaluer l'augmentation ou la perte de performance, nous avons besoin d'une mesure de la qualité d'un modèle.

6.1. Métriques d'évaluation

Un modèle A est considéré comme « meilleur » qu'un modèle B si les traductions produites par A sont de qualité supérieure à celles de B sur le même corpus. Historiquement, des humains annotaient manuellement la « qualité » de chaque traduction pour déterminer un score global. Cette façon de faire est très onéreuse, subjective, inconstante d'un évaluateur à l'autre et nécessite généralement plusieurs semaines. L'utilisation d'algorithmes d'évaluation automatique est donc favorisée. En plus d'être rapides, leurs résultats sont reproductibles, explicables et constants.

Cependant, réduire une mesure *qualitative* à une équation mathématique est une tâche périlleuse. Pour être utile, le score d'une métrique doit être fortement corrélé avec le jugement humain. Comme le démontre la grande variété d'algorithmes proposés ces dernières années (plus de 100 selon Marie *et al.* (2021)), il n'y a pas de consensus sur une *unique* métrique systématiquement en accord avec l'évaluation humaine. Marie *et al.* (2021) démontrent d'ailleurs que différents algorithmes peuvent produire des résultats contradictoires lorsque deux modèles sont comparés. Pour éviter une analyse biaisée et avoir une meilleure idée des performances d'un modèle, ils soulignent l'importance d'utiliser plus d'une mesure. Dans ce chapitre, nous en utilisons trois : BLEU (Papineni *et al.*, 2002), chrF++ (Popović, 2017) et BERTScore (Zhang *et al.*, 2019). Lorsque possible visuellement, nous afficherons côte à côte les résultats obtenus dans chaque métrique.

6.1.1. BLEU

La métrique BLEU, proposée par Papineni *et al.* (2002), est basée sur l'intuition que plus une traduction machine est « proche » d'une traduction humaine de référence, meilleure elle est. Pour une traduction candidate, BLEU produira donc un score de similarité avec la référence. Plusieurs références pour une même phrase candidate peuvent être fournies à l'algorithme pour lui permettre de reconnaitre, jusqu'à un certain point, les synonymes, les paraphrases, un ordonnancement différent, etc.

Le score est déterminé à partir du pourcentage de segments (de différentes tailles) de la phrase candidate qui se retrouvent dans une des phrases de référence. Mathématiquement, le score BLEU se calcule ainsi :

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{6.1.1}$$

Des segments de mots de taille 1-gramme jusqu'à N-grammes sont considérés. Pour chacun, un score de précision p_n est calculé à partir du ratio de n-grammes dans les phrases candidates qui se retrouvent dans une des phrases de référence associées. La précision est calculée sur tout le corpus, et p_n est techniquement une précision *modifiée* qui évite de recompter un segment déjà considéré. Toutes les précisions sont agrégées par une moyenne géométrique pondérée (en pratique, $w_n = 1/N$). Puisque le score est basé sur la précision, un modèle de traduction pourrait être tenté de produire de courtes phrases composées uniquement des mots les plus probables. La traduction obtiendrait ainsi une forte précision, même si elle ne traduit pas entièrement la phrase. Pour contrer ce problème, BLEU introduit un « *brievety penalty* » (BP) qui pénalise le score des traductions courtes. Cette pénalité est calculée sur tout le corpus à partir de la longueur des phrases candidates (c) et de celles des phrases de référence (r) :

$$\text{BP} = \begin{cases} 1 & \text{si } c > r \\ \exp(1 - r/c) & \text{si } c \leq r \end{cases} \tag{6.1.2}$$

Le score BLEU est un nombre entre 0 et 1, mais il est souvent rapporté sur 100 (ce que nous ferons dans notre mémoire). Un corpus traduit n'obtiendra généralement pas un score de 100, à moins d'être identique aux phrases de références. Il en découle que même une traduction humaine professionnelle atteindra rarement le score maximal.

Les avantages de cette métrique sont qu'elle est facile à calculer et qu'elle est corrélée avec le jugement humain (Papineni *et al.*, 2002). Elle est devenue, dans les deux dernières décennies,

la mesure la plus utilisée en TAN, Marie *et al.* (2021) rapportant un taux d'utilisation de 98,8 % parmi une sélection de 769 articles portant sur la traduction automatique.

La mesure a par contre ses problèmes. Elle n'est pas prévue pour être utilisée au niveau de la phrase, entre autres parce que les précisions n-grammes avec $n \geq 4$ valent souvent 0. Elle est également difficile à interpréter (que signifie un score de 32 ?). De plus, Callison-Burch *et al.* (2006) démontrent qu'une augmentation du score BLEU n'est pas nécessairement une indication d'une augmentation de la qualité. Finalement, en pratique, nous n'avons généralement pas accès à plusieurs références pour chaque phrase à traduire. Le score obtenu n'est donc pas flexible face à l'utilisation de synonymes, de paraphrases ou de variations morphologiques.

Dans notre mémoire, nous calculons les scores BLEU à l'aide de SacreBLEU [1] (Post, 2018).

6.1.2. chrF++

chrF++ (Popović, 2017) est une métrique basée sur chrF (Popović, 2015). Cette dernière, comme BLEU, calcule une similarité entre une traduction candidate et une ou plusieurs traductions de référence. Par contre, elle calcule un score-F qui prend en compte la précision *et le rappel*. De plus, ces deux valeurs sont calculées avec des segments de caractères et non de mots (ce ne sont pas des n-grammes de mots, mais des n-grammes de lettres). Ceci lui donne l'avantage d'être plus flexible aux variations morphologiques des mots. Par exemple, là où BLEU verrait en « manger » et « mangées » deux mots totalement différents, chrF y verrait une certaine similarité. De plus, la technique ne dépend pas de la stratégie de segmentation (« *tokenization* ») des mots.

Le score chrF se calcule ainsi :

$$\text{chrF}\beta = (1 + \beta^2)\frac{\text{chrP} \cdot \text{chrR}}{\beta^2 \cdot \text{chrP} + \text{chrR}} \tag{6.1.3}$$

Pour un n précis, `chrP` est le ratio de n-grammes de lettres de la phrase candidate qui se retrouvent dans la référence (la précision). `chrR` est le ratio des n-grammes de la référence qui se retrouvent dans la phrase candidate (le rappel). β est un paramètre qui permet de contrôler le poids du rappel par rapport à la précision. Un tel score est calculé pour chaque n et leur moyenne arithmétique donne le score chrF.

(Popović, 2017) démontre que de rajouter à cette moyenne un score chrFβ calculé sur les *mots* en plus de ceux calculés sur les caractères permet d'augmenter la corrélation avec le jugement humain. Elle propose donc chrF++ où le score est calculé par la moyenne des

1. Pour toutes nos expériences, la même configuration sera utilisée :
`nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0`

chrFβ sur les n-grammes de caractères où $n = \{1,2,3,4,5,6\}$ ainsi que des chrFβ sur les n-grammes de mots où $n = \{1,2\}$. Elle propose également $\beta = 2$.

Encore une fois, pour notre mémoire, nous utilisons SacreBLEU (Post, 2018) pour calculer les scores chrF++ [2] .

6.1.3. BERTScore

BERTScore (Zhang *et al.*, 2019) est une nouvelle métrique qui utilise les plongements contextuels pour comparer une phrase candidate et une phrase de référence. Pour deux phrases à comparer, BERTScore calcule les plongements des mots à partir d'un modèle BERT (Devlin *et al.*, 2018) préentrainé. La similarité cosinus entre chaque mot de la phrase candidate et chaque mot de la phrase de référence est calculée. Ceci détermine un alignement doux (« *soft* ») entre les mots. Ces valeurs d'alignement sont utilisées pour calculer une précision et un rappel, et le score-F1 est utilisé comme score BERTScore pour la phrase. Le score BERTScore d'un corpus est déterminé par la moyenne des scores de ses phrases. La métrique produit un nombre entre 0 et 1, mais nous le rapportons sur 100 dans notre mémoire.

Contrairement à BLEU et chrF++ qui comparent en surface les mots et les lettres utilisés, une mesure basée sur des plongements contextuels peut reconnaitre la similarité sémantique entre les mots. Ceci lui permet d'être stable face aux synonymes et aux paraphrases et ainsi de ne pas sous-estimer une phrase qui serait différente, mais sémantiquement semblable. Par exemple, pour la phrase de référence « *les gens aiment les voitures étrangères* », BLEU donnera un plus grand score à la phrase candidate « *les gens aiment voyager à l'extérieur* » plutôt qu'à la phrase « *la population préfère les voitures importées* », alors que cette dernière peut être considérée comme sémantiquement plus semblable (exemple inspiré de Zhang *et al.*, 2019). BERTScore a également l'avantage de ne pas dépendre de segments n-grammes. Ceci lui donne de la stabilité face à un ordonnancement différent du sujet et du complément (« Il est interdit d'utiliser des appareils électroniques » versus « L'utilisation d'appareils électronique est interdite »). Le plongement contextuel permet également de prendre en considération le rôle du mot dans la phrase.

Différentes configurations de la métrique sont possibles. Dans notre mémoire, nous utilisons la configuration par défaut de la librairie SentenceBERT [3] pour la langue française (`fr`). En particulier, les plongements sont calculés par le modèle « BERT multilingual base model (cased) » de HuggingFace [4] (Wolf *et al.*, 2020).

2. Pour toutes nos expériences, la même configuration sera utilisée :
`nrefs:1|case:mixed|eff:yes|nc:6|nw:2|space:no|version:2.0.0`
3. `https://www.sbert.net/`
4. `https://huggingface.co/bert-base-multilingual-cased`

6.1.4. Intervalle de confiance

En apprentissage automatique, nous nous intéressons à l'*espérance statistique* de la performance de notre modèle, c'est-à-dire la performance moyenne sur l'ensemble (possiblement infini) de toutes les phrases possibles d'un domaine. Il n'est évidemment pas possible de calculer exactement cette espérance, donc nous l'approximons sur un échantillon de phrases du domaine appelé « ensemble de test ». Mais un échantillon différent aurait-il donné un score différent ? Si oui, quelle aurait été l'amplitude de la variation ? Et lorsque nous comparons deux modèles, une différence de 0,5 est-elle significative ?

Pour répondre à ces questions, nous pouvons construire un intervalle de confiance d'un niveau γ ($0 \leq \gamma \leq 1$) autour du score d'un modèle. Que veut dire « niveau γ » pour un intervalle $[a,b]$? L'intervalle est calculé, à l'aide d'un algorithme, à partir de l'échantillon, et donc il peut varier d'un échantillon à l'autre. Si notre procédure pour calculer l'intervalle a une probabilité γ (probabilité calculée sur l'ensemble des échantillons possible) de produire des nombres a et b tels que le « véritable » score (l'espérance statistique du score) se trouve entre a et b, alors nous affirmons que $[a,b]$ est un intervalle de confiance associé à un niveau γ. Informellement, si γ est « grand », ceci nous permet de considérer $[a,b]$ comme un intervalle de valeurs plausibles pour le « véritable » score.

Dans les résultats de ce chapitre, nous utilisons toujours un niveau de confiance $\gamma = 0.95$. Nous parlerons alors « d'intervalle de confiance 95 % ».

6.1.4.1. Algorithmes pour calculer l'intervalle

Si le score d'un domaine est la moyenne des scores de chaque phrase (comme BERTScore) et que la distribution de ces derniers suit approximativement une loi normale, alors nous pouvons analytiquement calculer un intervalle de confiance pour le score du domaine. Pour un échantillon donné de phrases, $[\bar{x} - d,\bar{x} + d]$ sera un intervalle de confiance 95 % si $\bar{x}$ est la moyenne des scores des phrases et d est :

$$d = t \cdot \frac{s}{\sqrt{n}} \tag{6.1.4}$$

où s est l'écart-type empirique des scores des phrases, n est le nombre de phrases dans l'échantillon et t est un facteur qui dépend du niveau de confiance et de la valeur de n. Pour un niveau de 95 %, t vaut environ 1,96. Puisque la distribution des scores BERTScore des phrases suit approximativement une loi normale (confirmé empiriquement), nous pouvons utiliser cette procédure pour l'intervalle de confiance BERTScore.

BLEU n'est par contre pas une moyenne du score des phrases. Dans ces cas, nous déterminons un intervalle de confiance par une méthode de « *bootstrap resampling* » proposée par Koehn (2004). Un sous-ensemble de paires de phrases (1 000 par exemple) est sélectionné aléatoirement, avec remplacement. Le modèle traduit cet échantillon et un score est calculé. Nous répétons le processus un certain nombre de fois (400 fois par exemple). Nous obtenons ainsi une distribution de scores sur des corpus de taille 1 000. En déterminant le score au quartile 2,5 % et au quartile 97,5 %, nous pouvons définir un intervalle de confiance de niveau 95 %. Nous utilisons également cette stratégie avec les scores chrF++ et lorsque des scores BERTScore ont été calculés sur plusieurs échantillons du même domaine. Notons que Collins *et al.* (2005) soulèvent un doute quant à l'exactitude statistique de l'utilisation de la méthode de bootstrap pour estimer l'intervalle de confiance. Nous utilisons quand même la stratégie vu les résultats intéressants obtenus par Koehn (2004) et vu la facilité d'utilisation de la technique.

Dans nos graphiques, nous afficherons toujours la moyenne du score et son intervalle de confiance.

6.2. Résultats des modèles de comparaison

Pour rappel, nous avons trois modèles de comparaison. **`Base-WMT`** est un modèle Transformer (« big ») préentrainé, à l'aide de Fairseq (Ott *et al.*, 2019), sur 36 millions de paires de phrases anglais→français du corpus externe WMT14. **`Base-BTC`** est un modèle Transformer (« base ») que nous avons entrainé, à l'aide de Fairseq, sur un peu plus de 19 millions de paires fournies par le BTC. **Les six modèles affinés** (`Base-WMT-TEC`, `Base-WMT-CRI`, `Base-WMT-ENV`, `Base-BTC-TEC`, `Base-BTC-CRI` et `Base-BTC-ENV`) sont un des deux modèles de base (`Base-WMT` ou `Base-BTC`) affiné (à l'aide de Fairseq) sur les données d'entrainement du domaine le plus proche de chaque domaine de test (TEC : 1M de paires, CRI : 1,5M de paires, ENV : 1,3M de paires). L'appendice I détaille les configurations d'entrainement.

6.2.1. Résultats

La figure 6.1 présente les scores obtenus par chacun de ces modèles.

6.2.2. Discussion

Nous remarquons premièrement que, bien que `Base-WMT` ait une plus grande capacité et ait été entrainé sur plus de données, il obtient des scores inférieurs à `Base-BTC` sur les domaines FED et MEC. Cette différence n'est d'ailleurs pas unique à ces deux domaines. La figure 6.2 compare les scores BLEU de ces deux modèles sur tous les domaines du BTC. Nous y voyons

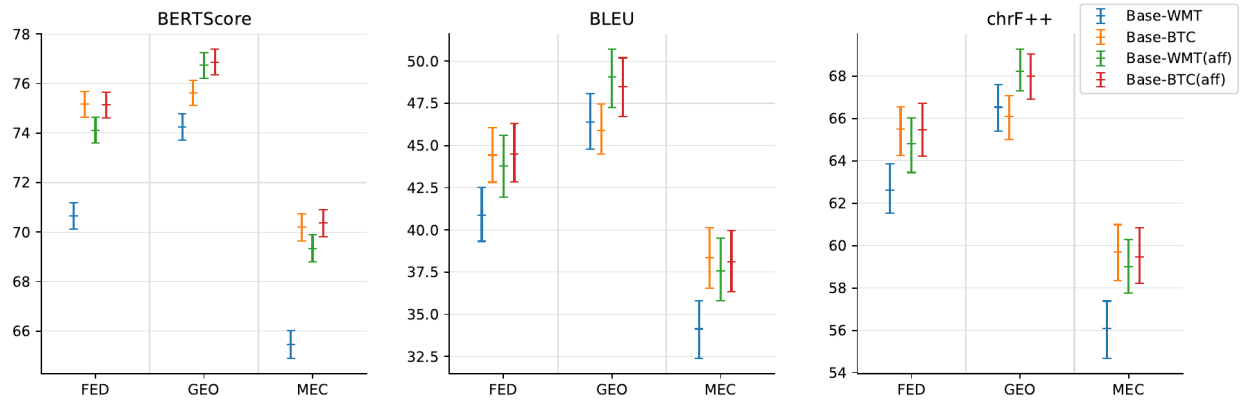

Figure 6.1. Scores des modèles de comparaison sur les trois domaines de test. Pour FED, le modèle appelé `Base-WMT(aff)` désigne le modèle affiné `Base-WMT-CRI`. Pour GEO et MEC, il désigne respectivement `Base-WMT-ENV` et `Base-WMT-TEC`. Même chose pour `Base-BTC(aff)`. Pour BERTScore, les intervalles de confiance 95 % sur la moyenne sont déterminés à l'aide de l'équation (6.1.4). Les intervalles de confiance 95 % de BLEU et chrF++ sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ».

que, à part sur le domaine GEO, les performances de `Base-BTC` sont systématiquement et significativement supérieures à celles de `Base-WMT` (le même effet est observé avec BERTScore et chrF++). Ces résultats ne sont pas surprenants. Comme expliqué au chapitre 4, les données utilisées pour l'entrainement des modèles « universels », comme `Base-WMT`, ne sont pas représentatives des données du BTC. Un modèle entrainé sur les données du BTC aura toujours un avantage sur un modèle universel et il sera plus adapté au style de traduction. Nous offrons à la section §6.6 une discussion sur l'importance des domaines utilisés pour l'entrainement.

Nous remarquons ensuite les performances supérieures des modèles affinés sur `Base-WMT` par rapport au modèle sans affinage (lignes vertes versus lignes bleues). Nous rappelons que les modèles affinés sont, pour chaque domaine de test, surentrainés sur le domaine d'entrainement le plus proche, et non sur le domaine de test (par exemple, pour le domaine de test FED, nous affinons sur le domaine CRI). Comparativement, pour `Base-BTC`, bien que l'amélioration du modèle affiné par rapport au modèle de base (lignes orange versus lignes rouges) soit intéressante sur GEO, elle est limitée, voire négative sur FED et MEC. Différents facteurs pourraient expliquer la situation. Premièrement, `Base-WMT` est un modèle avec une plus grande capacité que `Base-BTC`, préentrainé sur un plus grand corpus et pour plus longtemps. Il est probable qu'il peut plus facilement découvrir et modéliser les relations complexes entre les phrases sources et les phrases cibles d'un nouveau domaine. Deuxièmement, les données du BTC n'ont jamais été vues par `Base-WMT`. Chaque phrase contient une quantité d'informations nouvelles permettant au modèle de s'adapter. Pour `Base-BTC`, l'affinage lui présente des données qu'il a déjà vues et qui ne contiennent aucune nouvelle information. L'affinage

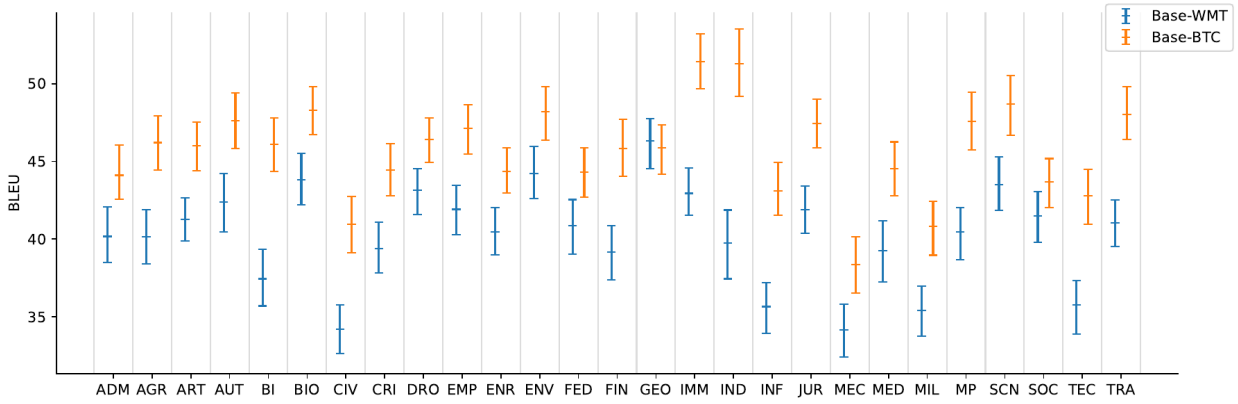

Figure 6.2. Comparaison des scores BLEU entre `Base-BTC` (orange) et `Base-WMT` (bleu) sur tous les domaines du BTC. Nous observons une performance supérieure de `Base-BTC` sur tous les domaines, à l'exception de GEO. Les domaines avec moins de 300 phrases dans leur ensemble de test (AQU, ELC, PAR, TEL) ont été ignorés, car le calcul d'un intervalle de confiance du score BLEU n'est pas stable avec aussi peu de phrases. Les intervalles de confiance 95 % sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ».

est alors plus un entrainement avec « emphase » sur certaines phrases plutôt que sur l'apport de nouvelles informations.

La stratégie d'affinage a cependant ses difficultés. Elle est très sensible au problème d'oubli catastrophique (voir section 2.2.1). Un modèle affiné sur un domaine perd de sa performance sur la majorité des autres domaines. Pour preuve, la figure 6.3 présente l'amélioration du score BLEU sur l'ensemble des domaines entre `Base-BTC` et le modèle affiné `Base-BTC-CRI`. Le modèle est affiné sur CRI puisqu'il s'agit du domaine le plus similaire à FED. Nous observons une amélioration sur le domaine d'affinage (CRI) de même que sur les domaines les plus proches (ADM et FED), mais les performances se dégradent sur tous les autres domaines. Nous avons également expérimenté avec les modèles `Base-BTC-ENV` et `Base-BTC-TEC`, et avons observé des résultats similaires : une amélioration significative sur 2 à 5 domaines, mais une dégradation significative sur les autres. Ceci implique que chaque modèle affiné n'est efficace que pour un nombre restreint de domaines, et donc plusieurs modèles affinés sont nécessaires pour obtenir une large couverture de l'espace des domaines. L'affinage est également très sensible au surapprentissage (« *overfitting* »). Nous avons expérimenté avec un modèle affiné sur FIN, plutôt que sur CRI (ces deux domaines étant les plus proches de FED). L'ensemble d'entrainement de FIN est relativement grand (790K paires), mais est environ la moitié de la taille de CRI (1,49M de paires). Cette différence a été suffisante pour que le modèle tombe rapidement en surapprentissage et qu'il ne réussisse pas à améliorer les performances sur un domaine autre que CRI. Des stratégies d'augmentation de l'ensemble

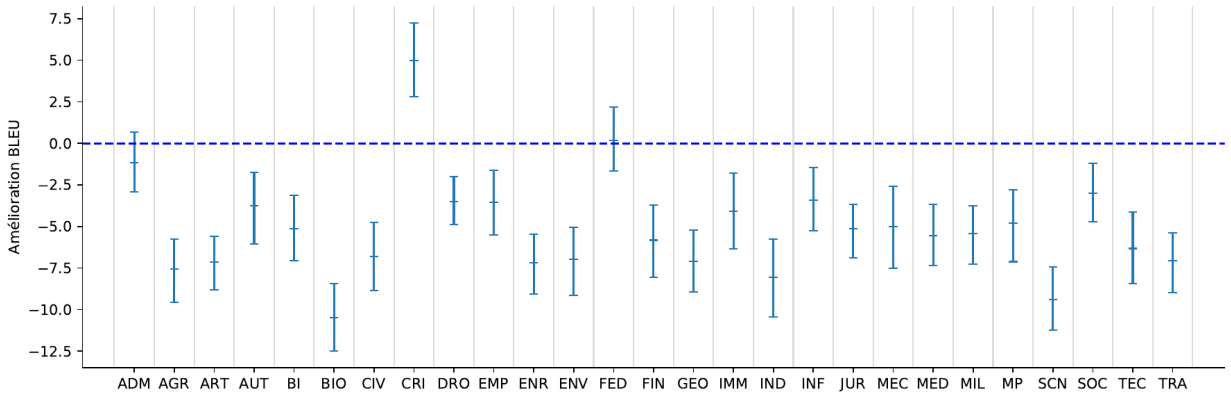

Figure 6.3. Amélioration du score BLEU entre `Base-BTC` et sa version affinée sur CRI (`Base-BTC-CRI`) sur tous les domaines. Si un score se trouve au-dessus de la ligne pointillée bleue, il y a amélioration du score, sinon il y a dégradation. À l'exception de ADM, CRI et FED, il y a oubli catastrophique sur tous les autres domaines. Les domaines avec moins de 300 phrases dans leur ensemble de test (AQU, ELC, PAR, TEL) ont été ignorés. Les intervalles de confiance 95 % sont déterminés en sélectionnant aléatoirement (par « *bootstrap* ») 400 échantillons de 1 000 phrases, et en calculant sur chacun la différence des scores obtenus par chaque modèle.

d'affinage et une recherche d'hyperparamètres seraient probablement nécessaires pour la création d'une variété de modèles affinés sur les domaines d'entrainement.

Toutes ces observations nous amènent à la conclusion que `Base-BTC` est le modèle le plus intéressant pour nos comparaisons. Ses performances dépassent globalement celles de `Base-WMT` et des modèles affinés sur `Base-WMT`. Les modèles affinés sur `Base-BTC` laissent entrevoir des performances théoriques intéressantes, mais en pratique, elles ne dépassent pas celles de `Base-BTC` sur deux des trois domaines, et leur complexité d'optimisation n'en fait pas des solutions réalistes.

Pour le reste de ce chapitre, nous allons principalement utiliser `Base-BTC` comme modèle de comparaison.

Notons que nos trois modèles de comparaison sont tous basés sur des architectures neuronales. Comme nous l'expliquons à la section §2.1, les modèles neuronaux atteignent généralement des performances supérieures aux architectures basées sur les modèles statistiques. Nous avons donc choisi nos modèles de comparaison, et leur configuration, afin qu'ils soient représentatifs des meilleurs modèles disponibles. Soulignons également que, même pour des scores plus faibles des métriques d'évaluation, les traductions qu'ils génèrent sont généralement considérées comme étant de bonne qualité.

6.3. Résultats de notre modèle

Pour rappel, notre modèle utilise une technique de plongement de domaine pour améliorer l'adaptation de domaine. Nous testons cinq techniques différentes — μ-`mots`, μ-`mots-contexte`, `sbert`, `pdd` et `tfidf`, et entrainons un modèle pour chaque technique. Dans le reste de ce chapitre, nous utiliserons le même nom pour désigner la technique et la version du modèle utilisant cette technique.

Le processus d'entrainement de chaque version du modèle est identique à celui de `Base-BTC`, à la différence qu'un plongement de domaine est rajouté à la phrase d'entrée (comme décrit à la section §3.1) et que le dictionnaire mots-plongements de l'encodeur est fixé (tel qu'expliqué à la section 3.2.6). Le même prétraitement des textes est appliqué, et le même vocabulaire de 40K mots est utilisé. Notons que les techniques de plongement nécessitent également une segmentation des phrases en mots. μ-`mots`, μ-`mots-contexte` et `pdd` utilisent la même segmentation BPE que `Base-BTC` ; `sbert` utilise la même stratégie de segmentation que Liu *et al.* (2019) ; et la stratégie pour `tfidf` est décrite à l'appendice F.

6.3.1. Stratégie d'évaluation

Au moment de traduire des textes d'un nouveau domaine, non vu durant l'entrainement, le modèle calcule premièrement le plongement de ce domaine avec sa technique associée. Ce plongement est calculé à partir de phrases extraites du domaine. Par exemple, si nous avons 100 phrases d'un nouveau domaine à traduire, nous pourrions toutes les utiliser pour calculer le plongement, et ensuite le fournir à notre modèle pour traduire ces mêmes phrases. Si nous recevons plus tard de nouvelles phrases du même domaine, nous pourrons réutiliser le même plongement précalculé.

Nous avons simulé cette situation pour tester nos modèles. Nous sélectionnons aléatoirement et avec remplacement, depuis le corpus du domaine, 100 phrases pour calculer son plongement [5] (nous expérimentons, à la section 6.4.1, avec d'autres tailles d'échantillons). Nous avons jugé que cette quantité de phrases (équivalent à cinq pages de ce mémoire) est représentative de la réalité de l'industrie lorsqu'un ensemble de textes d'un nouveau domaine est à traduire.

Une fois la représentation calculée, elle peut être employée pour traduire toute phrase du domaine, présente ou future (incluant celles utilisées pour le calcul du plongement). Nous

5. Nous soulignons cependant que ceci n'est que pour la phase de test. Durant l'entrainement, le corpus complet du domaine est utilisé pour calculer son plongement.

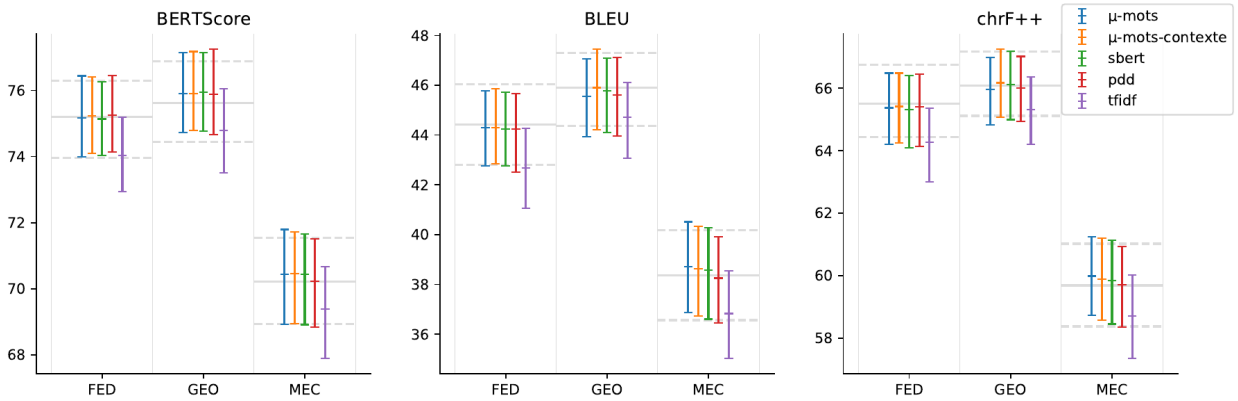

Figure 6.4. Scores de nos modèles sur les trois domaines de test. Chaque version de notre modèle utilise une des techniques de plongement de domaine. Nous présentons en gris pâle, derrière chaque domaine, le score moyen obtenu par Base-BTC (ligne pleine) et son intervalle de confiance (lignes pointillées). Les intervalles de confiance à 95 % sont déterminés par 20×20 (400) échantillons de 1 000 phrases sélectionnées par « *bootstrap* », comme expliqué à la section §6.3. La même procédure est également utilisée pour BERTScore.

simulons cette situation en évaluant chaque modèle sur un échantillon aléatoire (avec remplacement) de 1 000 phrases extraites du corpus du domaine. Une phrase peut donc se retrouver dans l'échantillon de 100 phrases et dans celui de 1 000 phrases.

Pour obtenir un intervalle de confiance, nous répétons 20 fois la procédure de sélection des 100 phrases pour le calcul du plongement. Pour chacun des plongements obtenus, nous répétons 20 fois la procédure de sélection et de traduction de 1 000 phrases. Nous obtenons ainsi 400 (20×20) corpus (traduits) de 1 000 phrases que nous évaluons avec nos métriques. Nous avons donc, pour chaque combinaison {métrique, domaine, technique de plongement}, 400 scores pour construire un intervalle de confiance.

6.3.2. Résultats

La figure 6.4 présente les scores obtenus par chaque modèle sur les trois domaines de test. Nous y présentons également, pour fin de comparaison, les scores obtenus par `Base-BTC` (lignes grises derrière les résultats).

6.3.2.1. Différences significatives

Un chevauchement entre l'intervalle de confiance d'un de nos modèles et de celui de `Base-BTC` n'implique pas nécessairement qu'il n'y a aucune amélioration ou dégradation (voir, par exemple, Frost, 2019). Pour chaque version de notre modèle, nous pouvons exécuter un test statistique sur la différence entre sa moyenne et celle de `Base-BTC`. La valeur-p obtenue

	BERTScore			**BLEU**			**chrF++**		
	FED	GEO	MEC	FED	GEO	MEC	FED	GEO	MEC
μ-mots	0,7609	**<1e-4**	**<1e-4**	0,9930	1,0000	**<1e-4**	0,9980	0,9998	**<1e-4**
μ-mots-contexte	0,3734	**<1e-4**	**<1e-4**	0,9876	0,5110	**<1e-4**	0,9689	**0,0161**	**<1e-4**
sbert	0,9492	**<1e-4**	**<1e-4**	0,9993	0,9938	**0,0007**	1,0000	0,2548	**0,0005**
pdd	0,1090	**<1e-4**	0,4340	0,9990	1,0000	0,9355	0,9864	0,9896	0,3547
tfidf	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tableau 6.1. Valeur-p du test-t de Welsh qui teste l'hypothèse que le score moyen obtenu par chaque version du modèle est inférieur ou égal à la moyenne obtenue par `Base-BTC`. Plus la valeur-p est petite, plus grande est la probabilité que notre modèle surpasse `Base-BTC`. Nous avons mis en évidence les scores inférieurs à 0,05. Les mêmes échantillons qu'à la figure 6.4 sont utilisés.

pourra nous informer s'il y a une différence statistique dans les performances. Le tableau 6.1 présente, pour chaque technique et sur chaque domaine, la valeur-p du test statistique [6] pour l'hypothèse nulle « *la moyenne du score du modèle est inférieure ou égale à celle du modèle de comparaison* `Base-BTC` ». Plus la valeur-p du test est faible, plus il est probable que la technique *améliore* le score.

6.3.2.2. Améliorations significatives

Ce test compare les moyennes, mais ne compare pas *l'amplitude* de leur différence. Même si un modèle a un score moyen significativement (d'un point de vue statistique) plus élevé que celui d'un autre, peut-on vraiment parler d'un « meilleur modèle » si cette différence n'est que de 0,01 BLEU ? Nous considérons qu'il y a réelle amélioration (ou dégradation) si la différence entre la moyenne des scores d'un modèle et celle de `Base-BTC` dépasse, en valeur absolue, un certain seuil. Nous proposons comme seuil 0,2 fois la déviation standard des 400 scores obtenus par `Base-BTC`. Par exemple, en BLEU, la déviation standard de `Base-BTC` est souvent environ 1, donc, dans ce cas, une différence sera significative à partir de 0,2 BLEU. Le tableau 6.2 présente les résultats. Un triangle vert indique une amélioration de la moyenne supérieure à ce seuil, un triangle inversé rouge indique une dégradation et un cercle gris indique que la différence ne dépasse pas ce seuil.

6.4. Analyse de notre modèle

Dans cette section, nous analysons les résultats de la section précédente pour évaluer la performance du modèle que nous proposons. Nous commençons par comparer les différentes

6. Nous utilisons le test t de Welch puisque les distributions des scores suivent approximativement une loi normale, mais que leur variance peut différer.

	BERTScore			**BLEU**			**chrF++**		
	FED	GEO	MEC	FED	GEO	MEC	FED	GEO	MEC
μ-mots	●	▲	▲	●	▼	▲	▼	▼	▲
μ-mots-contexte	●	▲	▲	●	●	▲	●	●	▲
sbert	●	▲	▲	▼	●	▲	▼	●	▲
pdd	●	▲	●	▼	▼	●	●	●	●
tfidf	▼	▼	▼	▼	▼	▼	▼	▼	▼

Tableau 6.2. Améliorations significatives des versions de notre modèle sur les domaines de test. ▲ : Le score entre `Base-BTC` et notre modèle a augmenté au-delà d'un certain seuil. ▼ : Le score a réduit au-delà d'un certain seuil. ● : La différence entre les scores ne dépasse pas un certain seuil. Le seuil vaut 0,2 fois la déviation standard de `Base-BTC` pour le score.

techniques de plongement. Nous poursuivons avec une analyse plus globale : le plongement apporte-t-il une information utile au modèle et permet-il d'améliorer les performances sur différents domaines ?

6.4.1. Comparaison des techniques de plongement

Avant de mesurer l'impact de l'incorporation du plongement dans le modèle, nous comparons d'abord les différentes techniques de plongement.

La technique `tfidf` offre de très mauvais résultats. Nous expliquons sa dégradation générale sur tous les domaines par la forte présence de bruit dans les vecteurs de plongement qu'elle génère. La technique construit le plongement en comptant, parmi les 100 phrases, le nombre d'occurrences de chaque mot d'un vocabulaire de plusieurs milliers d'entrées. Le vecteur qui en découle est donc extrêmement creux (« *sparse* » en anglais), la plupart des mots n'étant probablement pas présents dans ce petit ensemble de phrases. Deux ensembles du même domaine donneront probablement deux vecteurs complètement différents malgré leur source commune. Les plongements générés doivent donc être très variables et sensibles aux moindres variations dans le choix des mots, ce qui nuit au modèle.

La technique `pdd` n'offre généralement pas des performances significativement supérieures aux autres (en excluant `tfidf`). La technique tente d'extraire de chaque phrase une information concernant son domaine d'origine, mais peu d'informations sont sans doute disponibles dans les phrases que l'on pourrait qualifier de plutôt « génériques ». La technique est peut-être trop sensible à la présence de ces phrases dans l'ensemble utilisé pour calculer le plongement. Dans ce cas, l'information utile sur le domaine se trouverait sans doute noyée parmi des informations moins utiles. Ceci serait alors un obstacle à l'utilité du plongement, et causerait les pertes de performance observées dans certains cas.

Les techniques μ-mots, μ-mots-contexte et sbert semblent être les meilleures candidates pour notre tâche de traduction. En analysant la figure 6.4 et le tableau 6.2, nous remarquons qu'elles améliorent significativement, ou qu'elles gardent constantes, les performances sur la majorité des domaines dans toutes les métriques. Les deux premières techniques (μ-mots et μ-mots-contexte) utilisent des informations précalculées par Base-BTC sur les données du BTC. Ce transfert de connaissances (ou transfert d'apprentissage) leur permet sans doute d'être mieux adaptées au style des textes du BTC et ainsi de reconnaitre et d'encoder dans le plongement des informations utiles sur le domaine. Les deux dernières (μ-mots-contexte et sbert) utilisent des plongements de mots contextualisés dans leur calcul du plongement. Cela leur permet probablement de produire des plongements qui encodent une information supplémentaire sur la structure des phrases et le rôle sémantique des mots, permettant au modèle de mieux reconnaitre les différences entre les domaines. μ-mots-contexte est la technique qui intègre ces deux aspects et elle semble effectivement surpasser les deux autres. C'est la seule dont les performances ne se dégradent significativement sur aucun des domaines dans toutes les métriques.

À partir des observations précédentes, nous proposons les hypothèses suivantes sur ce qui constitue une bonne technique de plongement de domaine pour notre tâche de traduction. Premièrement, elle encode dans le plongement généré une information qui va au-delà de la simple présence ou absence d'un mot. Par l'utilisation de plongements contextuels, deux des meilleures techniques réussissent probablement à encoder une information sur la structure des phrases, ce qui peut permettre au modèle de mieux discerner les domaines. Deuxièmement, la technique a avantage à être adaptée aux données qui seront utilisées pour l'entrainement du modèle de traduction. Par exemple, μ-mots-contexte utilise les plongements calculés par un Transformer entrainé sur ces données. Le modèle de traduction reconnaitra ainsi probablement mieux les relations entre les signaux présents dans le plongement et les connaissances acquises sur les données d'entrainement. Ceci laisse par contre supposer l'importance du choix des domaines utilisés pour l'entrainement, et nous proposons une discussion à ce sujet à la section §6.6.

Nous soulignons par contre que la qualité des plongements dépend probablement du nombre de phrases utilisées pour le calculer. Comme nous l'avons expliqué à la section 6.3.1, nous utilisons toujours dans nos tests un ensemble de 100 phrases pour calculer le plongement du domaine de test. Cependant, peut-être qu'une taille différente aurait donné des résultats significativement différents de ceux observés précédemment.

Nous avons reproduit les expériences de la section §6.3, mais en variant la taille de l'échantillon utilisé pour le calcul du plongement. La figure 6.5 présente les scores BERTScore obtenus (les scores BLEU et chrF++ de ces expériences sont présentés à l'appendice M).

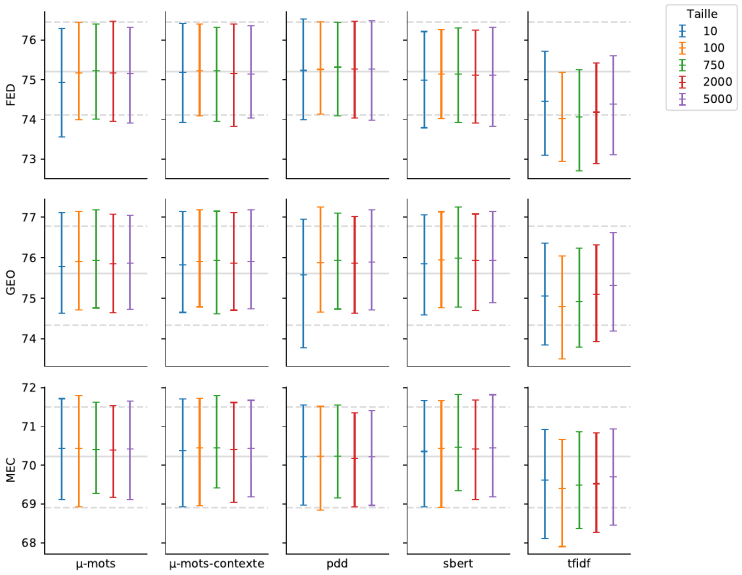

Figure 6.5. Évolution du score BERTScore selon la taille d'échantillon utilisée pour calculer le plongement. Pour chaque paire modèle/domaine, nous utilisons diverses tailles d'échantillons de phrases du domaine (10, 100, 750, 2000 et 5000) pour calculer le plongement. Pour la plupart des modèles, nous remarquons une légère augmentation de la performance avec l'augmentation de la taille d'échantillon utilisée, jusqu'à un plafonnement à 750. Notons tout de même que les performances des plus petits échantillons restent globalement semblables à celles des plus gros. Les intervalles de confiance 95 % sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ». Les scores BLEU et chrF++ sont présentés à l'appendice M

En nous concentrant plus particulièrement sur les trois meilleures techniques (`μ-mots`, `μ-mots-contexte` et `sbert` ; colonnes 1, 2 et 4 respectivement) nous remarquons une légère amélioration avec la taille, jusqu'à un plafonnement après 750. Nous expliquons ce phénomène par la variance des plongements calculés sur peu de phrases (nous remarquons d'ailleurs parfois un plus grand intervalle de confiance sur les plus petites tailles). Au-delà d'un certain nombre de phrases, le vecteur semble se stabiliser.

Par contre, ces améliorations restent mineures. Nous en tirons la conclusion que les techniques de plongement peuvent bénéficier d'un plus grand échantillon de phrases, mais qu'il n'y a pas d'intérêt à dépasser quelques centaines. De plus, les performances restent bonnes (par rapport au maximum pour la technique) même avec des ensembles entre 10 et 100 phrases. Ceci est un avantage important du modèle et des techniques de plongement que nous proposons : une adaptation est possible avec seulement un petit ensemble de phrases à traduire.

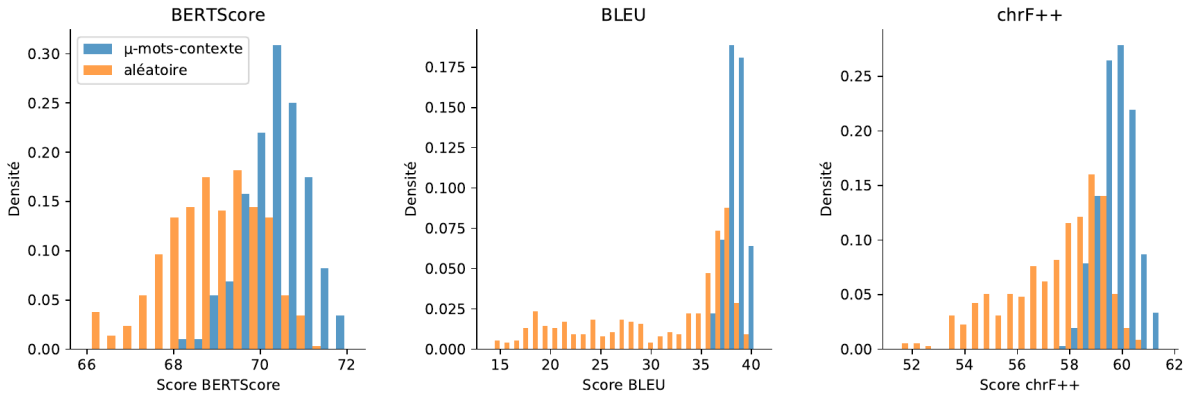

Figure 6.6. Distributions de 400 scores obtenus par μ-`mots-contexte` (en bleu) et une technique de plongement aléatoire (en orange) sur le domaine MEC. Nous remarquons une distinction claire entre les distributions, et un déséquilibre dans ceux de la technique `aléatoire`.

6.4.2. Évaluation de l'utilité du plongement

Notre modèle est structurellement très semblable à `Base-BTC`, la seule différence est l'accès à une information sur le domaine. Les résultats sont également semblables. Nous pourrions donc raisonnablement nous demander s'il y a un avantage réel à notre modèle. Pour nous éclairer, nous analysons plus en détail, dans cette section, l'impact du plongement et son utilisation par notre modèle.

Une première question est de savoir si le modèle utilise réellement le plongement, ou si celui-ci n'est considéré, par le modèle, que comme un bruit à ignorer.

Si le modèle n'utilise effectivement pas le plongement, alors un modèle qui aurait été entrainé avec des plongements aléatoires [7] et qui utilise des plongements aléatoires pour tout nouveau domaine ne devrait pas obtenir des scores significativement différents d'avec un de nos modèles. Nous avons fait l'expérience et la figure 6.6 compare la distribution des scores obtenus par cette technique (appelée « `aléatoire` ») à ceux obtenus par μ-`mots-contexte`.

Nous y remarquons une claire dégradation du score moyen lorsque la technique `aléatoire` est utilisée. Plus précisément, pour certaines phrases, le modèle réussit à produire des traductions de qualité, mais pour d'autres, la qualité n'y est pas. Nous avons visuellement analysé les mauvaises traductions et l'appendice J présente quelques exemples. Nous avons observé que, généralement, les premiers mots sont correctement générés, mais le modèle semble trébucher lorsqu'il en atteint certains. Nous émettons l'hypothèse que, pour ces mots, le modèle requiert l'information qui se trouve dans le plongement. Cependant, plusieurs autres phrases ont pu

7. Notons que l'algorithme aléatoire que nous avons utilisé s'assure de produire le même vecteur pour un même ensemble de textes. Le même vecteur est donc toujours généré pour le même domaine.

être traduites correctement et dans leur entièreté. Le modèle ne semble donc pas toujours avoir besoin du plongement pour traduire une phrase. Il parait porter une attention variable sur ce plongement d'un mot à l'autre et d'une phrase à l'autre.

Pour valider cette théorie, nous avons analysé l'attention que l'encodeur du modèle porte sur le plongement (le premier mot) pour chacun des mots de deux phrases. Nous rappelons que dans l'encodeur d'un Transformer, à chaque couche (six dans notre cas), un module calcule, pour chaque mot, une valeur d'attention envers chacun des autres mots. Par exemple, dans la phrase *« Jean lance la balle et son chien la lui rapporte »*, le modèle calculera probablement, pour le mot « *lui* », une plus forte attention sur le mot « *Jean* » que sur le mot « *balle* », alors que pour le deuxième « *la* », ça serait l'inverse. Si l'encodeur utilise plusieurs têtes (huit dans notre cas), une valeur d'attention sera calculée pour chaque tête. La figure 6.7 présente, pour deux phrases différentes, le poids moyen d'attention (sur toutes les têtes) portée par chaque mot sur celui du plongement (c'est-à-dire le premier mot ; non affiché dans la figure). Nous présentons cette attention pour chacune des six couches.

Nous voyons que l'attention varie d'un mot à l'autre : le plus d'attention est portée par le quatrième mot dans la première phrase, alors qu'il s'agit du sixième dans la seconde phrase. L'attention globale n'est pas non plus égale d'une phrase à l'autre, étant plus forte dans la première phrase que dans la deuxième.

Les expériences précédentes tendent à démontrer que le modèle utilise effectivement le plongement du domaine lorsqu'il génère des traductions. Le niveau d'attention — ou de façon équivalente, le niveau « d'utilité » du plongement — est cependant variable. Le plongement semble même, parfois, non essentiel pour générer une traduction valide.

Le modèle utilise le plongement, mais utilise-t-il l'information descriptive du domaine qui s'y trouve ? Pour reprendre l'expression que nous avons introduite à la section §3.1, le modèle reconnait-il la « signature de signaux » ?

Notre modèle est entrainé avec seulement 25 plongements de domaines différents. Il est probable que le modèle les considère comme simplement 25 identifiants uniques et indépendants, sans chercher à profiter de l'information qui s'y trouve pour adapter ses traductions. Nous avons d'ailleurs vu à la section 2.2.1 une technique multidomaines semblable : un mot pour chaque domaine est rajouté au début de chaque phrase et le modèle les utilise comme identifiant.

Pour tester si le modèle utilise l'information du domaine qui se trouve dans le plongement, nous proposons une expérience où le même ensemble de phrases est traduit plusieurs fois à l'aide du plongement de différents domaines. Nous nous attendons, bien sûr, à une diminution du score si le plongement d'un autre domaine que celui d'où proviennent les textes est utilisé. Cependant, si le modèle ne porte effectivement pas attention à l'information qui se

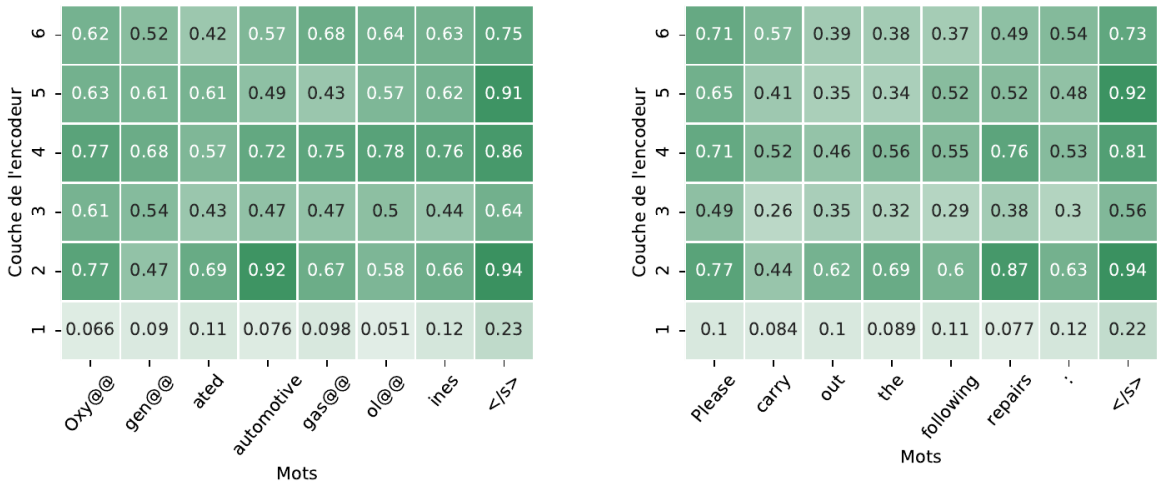

Figure 6.7. Attention de l'encodeur portée sur le premier mot (celui du plongement) par tous les autres mots de deux phrases. Les deux phrases sont « `Oxygenated automotive gasolines` » et « `Please carry out the following repairs:` ». Les phrases sont segmentées en différents mots par BPE (qui peut parfois diviser des mots en sous-mots comme « `Oxygenated` » qui devient les trois mots « `Oxy@@` », « `gen@@` » et « `ated` »). Le mot-étiquette contenant le plongement (le premier mot) *n'est pas affiché*, mais, en pratique, il se retrouve au début de chacune de ces phrases lorsque le modèle les traduit. Nous affichons l'attention moyenne (sur toutes les têtes) pour chaque couche et chaque mot. Par exemple, pour le deuxième mot de la première phrase (`gen@@`), le module d'attention calculera un poids d'attention envers chacun des autres mots de la phrase, y compris le mot-étiquette (non affiché). Ces poids, tous positifs, somment à 1. Ce tableau ne montre que les poids d'attention envers le mot-étiquette, et ceci pour les modules d'attention de chaque couche de l'encodeur. Nous utilisons le modèle μ`-mots-contexte` avec le plongement calculé sur les phrases d'entrainement du domaine MEC.

trouve dans le plongement, cette diminution de score devrait être semblable, peu importe le plongement utilisé. Par exemple, si nos textes proviennent du domaine MEC, nous nous attendons à voir sensiblement les mêmes résultats que nous utilisions le plongement du domaine TEC, CRI ou JUR. Par contre, si le modèle utilise l'information, nous devrions voir une dégradation du score corrélée avec la perte de similarité entre le domaine d'origine et celui utilisé pour la traduction. Par exemple, la perte serait plus petite en utilisant le plongement de TEC que celui de JUR puisque ce dernier est moins similaire à MEC.

L'expérience se déroule ainsi : pour chaque domaine de test, nous sélectionnons aléatoirement un ensemble de phrases [8]. Nous construisons ensuite six plongements, calculés à partir des textes de six domaines : le domaine de test lui-même, les deux domaines les plus similaires (selon la mesure de similarité de la section §4.4), les deux domaines les moins similaires et

8. La même procédure par *bootstrap* que dans la section §6.3 est utilisée.

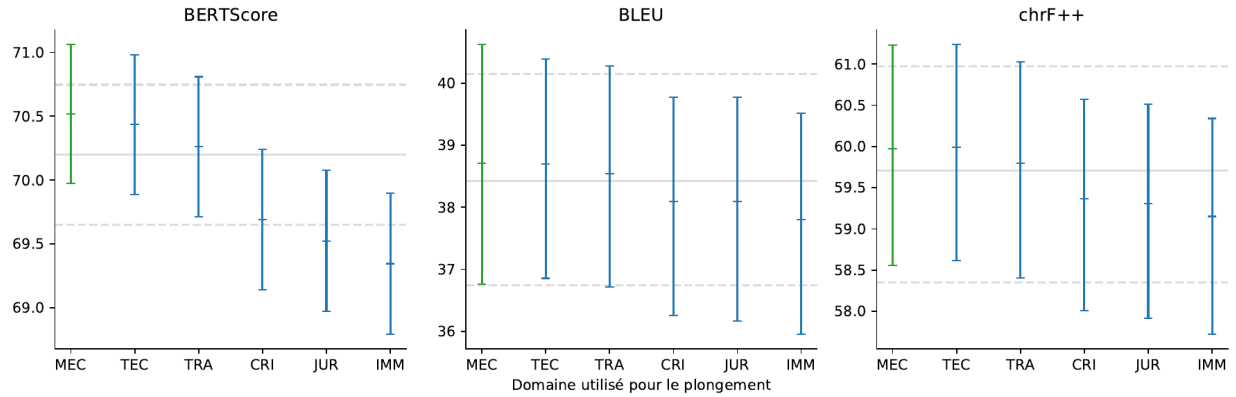

Figure 6.8. Évolution du score sur des textes du domaine MEC lorsque le plongement d'un autre domaine (axe horizontal) est utilisé. Le premier intervalle (vert, à gauche de chaque graphique) montre le score obtenu lorsque le plongement du même domaine que les textes est utilisé. Le deuxième intervalle présente le score obtenu lorsque le plongement du domaine le plus proche est utilisé (sur les mêmes textes). Plus un domaine est à droite, moins il est similaire avec MEC. Nous remarquons une claire dégradation avec la perte de similarité du domaine. En gris, en arrière, nous affichons le score (et son intervalle de confiance) obtenu par `Base-BTC`. Le modèle μ`-mots` est utilisé pour générer les traductions. Les intervalles de confiance 95 % sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ». L'appendice K présente les résultats obtenus sur les deux autres domaines de test.

un domaine entre les deux. L'ensemble de phrases est ensuite traduit six fois en utilisant chacun des six plongements calculés.

La figure 6.8 présente les résultats de cette expérience sur le domaine MEC avec la technique de plongement μ`-mots`. La figure affiche premièrement, en vert, le score obtenu lorsque nous utilisons le plongement calculé pour le domaine MEC. Sont ensuite affichés les scores obtenus lorsque nous utilisons le plongement des cinq autres domaines. Les domaines sont triés en ordre décroissant de similarité (le plus similaire à gauche).

Il y a une corrélation claire entre le score et la similarité du domaine. Ceci démontre deux choses : premièrement, la technique de plongement a réussi à produire une représentation qui intègre une information permettant de situer le domaine par rapport à un autre. Deuxièmement, nous voyons que le modèle est capable d'interpréter cette information pour guider ses traductions. Nous avons refait cette expérience avec les deux autres domaines de test (FED et GEO) et avons obtenu des résultats semblables (voir appendice K).

Nous souhaitions évaluer visuellement l'effet de cette variation du domaine sur les traductions générées. Par exemple, remarque-t-on des changements clairs lorsque le plongement du domaine ENV est utilisé au lieu de celui de SCN ?

Source	*The use of electronic devices, with the sound turned on, during classes is strictly prohibited.*
Réf.	Il est formellement interdit d'utiliser des appareils électroniques qui peuvent émettre un bruit en classe.
Base-BTC	L'utilisation d'appareils électroniques, avec le son allumé, pendant les classes est strictement interdite.
GEO	L'utilisation d'appareils électroniques, avec le son allumé, pendant les classes est strictement interdite.
ENV	Il est strictement interdit d'utiliser des appareils électroniques, avec le son allumé, pendant les classes.
IMM	Il est strictement interdit d'utiliser des appareils électroniques, avec le son allumé, pendant les cours.
JUR	L'utilisation d'appareils électroniques, avec le son allumé, pendant les classes est strictement interdite.
MP	Il est strictement interdit d'utiliser des appareils électroniques, avec le son allumé, pendant les classes.
SCN	Il est strictement interdit d'utiliser des appareils électroniques, avec le son allumé, pendant les classes.

Figure 6.9. Exemple représentatif de traductions obtenues en variant le domaine utilisé pour le plongement. La phrase est tirée du domaine GEO. Après la phrase source originale (Source) et la traduction humaine de référence (Réf.), nous avons la traduction générée par `Base-BTC` suivie des traductions obtenues par μ`-mots-contexte` en utilisant le plongement calculé sur divers domaines (l'ensemble d'entrainement complet du domaine est utilisé pour le plongement). Le premier (GEO) est le plongement calculé sur le même domaine que celui de la phrase.

Après avoir analysé environ 200 traductions dans les différents domaines de test, nous n'avons pas réussi à observer des caractéristiques claires et constantes entre les plongements des domaines utilisés. Il y avait variation dans les traductions produites, mais elles ne permettaient pas de reconnaitre des traits caractéristiques pour chaque domaine. Par exemple, pour certaines phrases, utiliser le plongement de ENV occasionne de petites variations, mais elles ne peuvent pas être distinguées de celles qui auraient été produites en utilisant le plongement de SCN.

La figure 6.9 montre un exemple d'une phrase source et des différentes traductions obtenues en variant le plongement utilisé. Cet exemple est typique de ceux rencontrés lors de notre analyse.

Nous supposons que l'impact du domaine utilisé se fait significativement ressentir uniquement sur de grands corpus, et qu'il ne se présente pas clairement sur un petit ensemble de textes, comme celui que nous avons analysé.

6.4.3. Performances sur d'autres domaines

L'objectif que nous poursuivons est de créer un modèle capable de s'adapter à tout domaine, et d'y obtenir des performances supérieures à un modèle entrainé de façon traditionnelle, comme `Base-BTC`.

Nous avons jusqu'à présent évalué notre modèle sur trois domaines tirés des données du BTC et non vus à l'entrainement. Nous étendons dans cette section l'analyse à d'autres domaines.

Nous évaluons premièrement les performances du modèle sur les domaines utilisés pour l'entrainement. Ce test est important puisqu'il nous permet de déceler s'il y a oubli catastrophique. L'oubli catastrophique, tel que nous l'avons défini à la section 2.2.1, se manifeste lorsque les performances d'un modèle sur un domaine A se dégradent lorsque nous l'adaptons à un domaine B. Dans cette section, nous utiliserons plutôt cette expression si le score de notre modèle sur un domaine baisse significativement par rapport au score obtenu par `Base-BTC`. « Significativement » implique ici qu'il y a perte au-delà d'un certain seuil.

Il est important d'éviter le plus possible la dégradation sur les domaines utilisés *pour l'entrainement*. Le modèle doit être efficace sur tous les domaines, car il pourra être appelé à traduire de nouveaux textes de n'importe quel domaine.

Nous avons testé nos deux meilleurs modèles (μ`-mots-contexte` et `sbert`) ainsi que `pdd`. Les ensembles de test sont construits à partir de phrases qui ont été *exclues* des données d'entrainement (voir la section §4.5). La figure 6.10 présente les scores BERTScore obtenus sur les différents domaines [9]. Les scores BLEU et chrF++ sont présentés à l'appendice L.

Pour donner plus de sens à ces scores bruts, nous présentons dans la figure 6.11 le nombre de domaines sur lesquels il y a dégradation ou amélioration significative (rangées (a) et (b) respectivement) entre nos modèles et `Base-BTC`. Un changement est considéré comme significatif si la différence excède un certain seuil de notre choix. Les graphiques présentent donc le nombre de changements significatifs en fonction du seuil choisi. Par exemple, pour le graphique des dégradations en BLEU (graphique en au haut, au centre), si au seuil 0,2 un modèle a une valeur de 3, ceci indique qu'il a une dégradation d'au moins 0,2 BLEU sur 3 domaines. Autre exemple, pour le graphique des améliorations en chrF++ (graphique du bas, à droite), si au seuil 0,1 un modèle a une valeur de 13, ceci indique qu'il a une amélioration d'au moins 0,1 chrF++ sur 13 domaines.

Un bon modèle aurait une courbe de dégradations (rangée (a)) qui descend rapidement et une courbe d'améliorations (rangée (b)) qui reste élevée le plus longtemps possible.

9. Pour obtenir des intervalles de confiance stables, nous avons exclu les six domaines avec moins de 700 phrases dans leur ensemble de test (AQU, ART, ELC, IND, PAR, TEL)

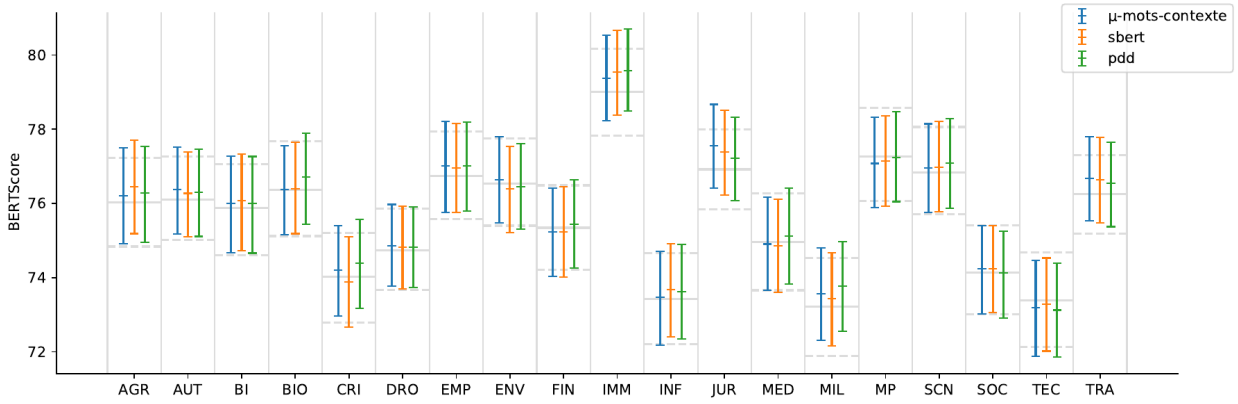

Figure 6.10. Scores BERTScore de trois de nos modèles sur les domaines d'entrainement. Pour chaque domaine, nous affichons également en gris le score moyen et l'intervalle de confiance du modèle `Base-BTC` sur ce domaine. Les domaines avec moins de 700 phrases dans leur ensemble de test ont été exclus (AQU, ART, ELC, IND, PAR, TEL). Les intervalles de confiance 95 % sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ». L'appendice L présente les résultats pour BLEU et chrF++.

Selon ces critères, le meilleur modèle serait `pdd` (ligne verte dans la figure 6.11). Ceci est étonnant puisque ses performances n'ont pas été jugées intéressantes sur les trois domaines de test à la section 6.4.1. Il semble donc que ce modèle soit tombé dans une sorte de surapprentissage : la technique de plongement est trop spécialisée dans la création de représentations pour les domaines d'entrainement, et celles qu'elle produit pour d'autres domaines ne contiennent pas une information aussi utile.

Pour les deux autres modèles, du point de vue des dégradations significatives (rangée (a)), μ`-mots-contexte` (ligne bleue) présente de meilleures performances que `sbert` (ligne orange). Ses courbes descendent plus rapidement, donc pour un même seuil, elle souffre moins d'oubli catastrophique. Du côté des améliorations significatives (rangée (b)), les deux techniques ont en général des résultats semblables. μ`-mots-contexte` semble donc, encore une fois, être une technique globalement supérieure.

Pour nos deux meilleures techniques, nous tirons les conclusions suivantes. Elles peuvent souffrir d'oubli catastrophique sur certains domaines. Cependant, le nombre de domaines où il y a dégradation significative est relativement faible par rapport au nombre de domaines sans dégradation. Pour la majorité des seuils « raisonnables » (ex. : 0,2 BLEU), nous comptons entre 1 et 3 oublis sur 19 domaines testés. De plus, pour ces mêmes seuils, nous observons entre 6 et 10 améliorations significatives. Nos modèles utilisant ces techniques semblent donc montrer une performance globale significativement supérieure au modèle de comparaison `Base-BTC`. Nous devons par contre rester prudents et surveiller les oublis catastrophiques.

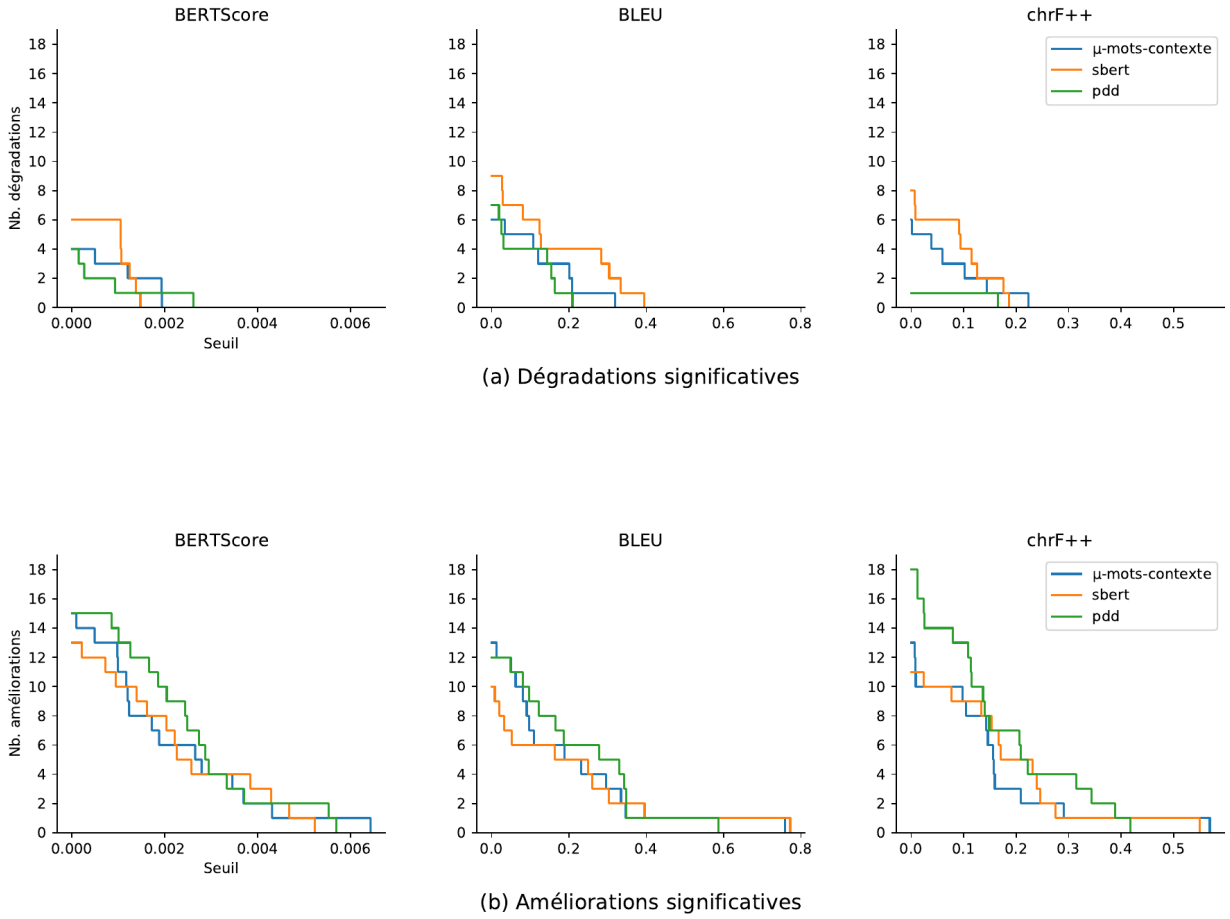

Figure 6.11. Nombre de domaines sur lesquels il y a dégradation (a) ou amélioration (b) significative du score en fonction du seuil de signification utilisé. Pour les dégradations (a), une meilleure technique aura une courbe qui descend rapidement. Pour les améliorations (b), une meilleure technique aura une courbe qui reste élevée le plus longtemps possible. Nous avons, au total, 19 domaines.

Nous présentons à l'appendice L, en plus des graphiques des scores BLEU et chrF++, des tableaux indiquant les domaines précis sur lesquels nous avons observé une amélioration ou une dégradation significative pour un seuil fixé.

Les domaines que nous avons utilisés jusqu'à présent sont tous extraits de la même source, le BTC. Ceci nous a permis d'évaluer notre modèle sur des données réelles de l'industrie. Nous souhaitons tout de même tester notre modèle sur des données extérieures au BTC et donc, à priori, fort différentes de celles vues jusqu'à présent.

Nous avons testé trois de nos modèles (`μ-mots-contexte`, `sbert` et `pdd`) sur cinq nouveaux domaines bilingues et parallèles [10].

10. Le corpus « COVID-19 » peut être téléchargé à `https://tico-19.github.io/memories.html`, les corpus des autres domaines peuvent être téléchargés à `https://opus.nlpl.eu/index.php`.

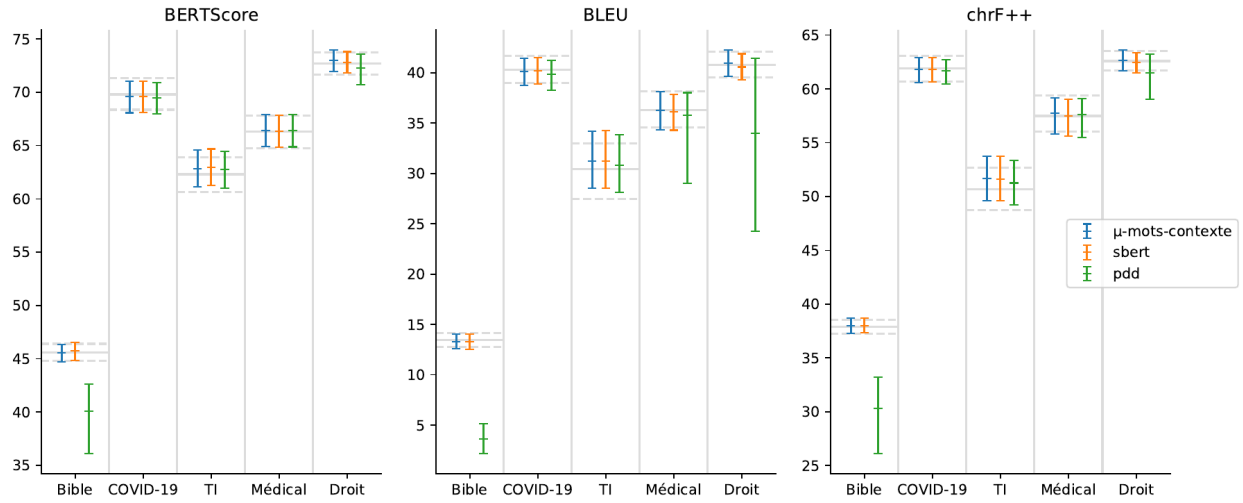

Figure 6.12. Scores de trois de nos modèles sur des domaines extérieurs au BTC. Pour chaque domaine, nous affichons également en gris le score moyen et l'intervalle de confiance du modèle `Base-BTC` sur ce domaine. Les intervalles de confiance 95 % sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ».

Bible Traductions de la Bible (Tiedemann, 2012; Christodouloupoulos et Steedman, 2015). Une analyse rapide de ce corpus fait ressortir la forte différence de style d'avec le corpus du BTC.

COVID-19 Corpus de textes reliés à la COVID-19 (Anastasopoulos *et al.*, 2020).

TI Textes des technologies de l'information (OpenOffice, PHP, GNOME, KDE, Ubuntu) (Tiedemann, 2012).

Médical Textes médicaux de l'*European Medicines Agency* (Tiedemann, 2012).

Droit Textes législatifs de l'Union européenne (Tiedemann, 2012).

Nous appliquons les mêmes étapes de prétraitement qu'à la section §4.3 et utilisons la même stratégie de test qu'à la section §6.3. La figure 6.12 présente les scores obtenus et, comme à la section 6.3.2.2, nous présentons dans le tableau 6.3 les améliorations (ou dégradations) significatives entre nos modèles et `Base-BTC`.

Les performances de μ`-mots-contexte` semblent encore une fois significativement supérieures à `sbert`. En particulier, μ`-mots-contexte` est la seule technique à améliorer « Droit ». Les performances de `pdd` sont très variables. Nous avons cependant précédemment émis l'hypothèse que `pdd` est probablement tombé en surapprentissage sur les domaines d'entrainement, et que le même phénomène explique ici ses faibles performances.

Pour les deux meilleures techniques, nous notons en général une dégradation significative sur 1 ou 2 domaines, et une amélioration ou un maintien sur 3 ou 4 domaines. Ces observations nous amènent encore une fois à la conclusion que le modèle que nous proposons semble

	BERTScore					BLEU					chrF++				
	Bible	COVID-19	TI	Droit	Médical	Bible	COVID-19	TI	Droit	Médical	Bible	COVID-19	TI	Droit	Médical
μ-mots-contexte	●	▼	▲	▲	●	▼	▼	▲	▲	●	▲	●	▲	●	▲
sbert	▲	▼	▲	●	●	▼	●	▲	▼	●	▲	●	▲	▼	●
pdd	▼	▼	▲	▼	●	▼	▼	▲	▼	▼	▼	▼	▲	▼	●

Tableau 6.3. Améliorations significatives du score de trois de nos modèles par rapport à `Base-BTC` sur des domaines extérieurs au BTC. ▲ : Le score entre `Base-BTC` et le modèle a augmenté au-delà d'un certain seuil. ▼ : Le score a réduit au-delà d'un certain seuil. ● : La différence entre les scores ne dépasse pas un certain seuil. Le seuil vaut 0,2 fois la déviation standard de `Base-BTC` pour le score.

offrir des performances globalement égales ou supérieures à un modèle tel `Base-BTC`. Nous soulignons par contre encore une fois une certaine tendance à l'oubli catastrophique.

Les gains et les dégradations observés sont plus prononcées sur certains domaines. Toutes les techniques ont amélioré les performances sur « TI », et toutes semblent avoir de la difficulté sur « COVID-19 ». Nous croyons que ces forces et ces faiblesses partagées s'expliquent par leur point commun : les données d'entrainement. Nous proposons, à la section §6.6, une discussion sur l'importance du choix des domaines pour l'entrainement du modèle.

6.5. Comparaison avec d'autres solutions

6.5.1. Modèles de comparaison

Comme nous l'avons déjà démontré à la figure 6.2, les performances de `Base-WMT` sont significativement inférieures à celles de `Base-BTC` sur tous les domaines du BTC, à l'exclusion de GEO. Elles sont, par le fait même, inférieures à celles obtenues par les différentes versions du modèle que nous proposons. Cependant, ses performances sont significativement supérieures à nos modèles sur les cinq domaines extérieurs. En effet, la figure 6.13 présente les mêmes résultats qu'à la figure 6.12 auxquels nous avons rajouté les résultats obtenus par `Base-WMT` (en rouge). Nous y observons une supériorité totale de `Base-WMT` sur les autres modèles. Ce modèle peut donc sembler supérieur en général. Nous expliquons ses performances par une architecture plus complexe (le triple des paramètres) et une stratégie d'entrainement plus complète (le double des données d'entrainement, un entrainement sur plus d'époques). Cependant, comme nous avons démontré qu'à architecture et données

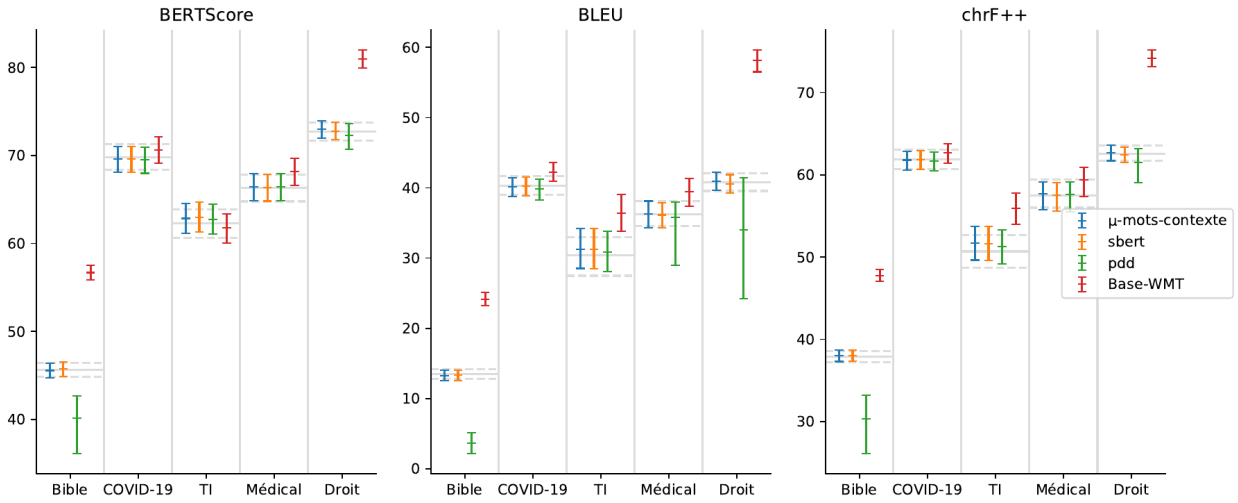

Figure 6.13. Reprise des scores de nos modèles sur les domaines extérieurs (figure 6.12) auxquels nous avons rajouté les scores obtenus par `Base-WMT` (en rouge).

d'entrainement égales notre modèle est globalement plus performant qu'un modèle traditionnel (c'est-à-dire `Base-BTC`), nous supposons que les mêmes gains par rapport à `Base-WMT` auraient été observés si nous avions utilisé la même architecture et les mêmes données d'entrainement. Soulignons par contre l'avantage de `Base-WMT` : ce modèle est préentrainé, il ne requiert aucun processus d'entrainement supplémentaire.

Concernant `Base-BTC`, ses bons résultats démontrent qu'un modèle de traduction traditionnel entrainé sur toutes les données est déjà un modèle performant. Cette observation avait déjà été faite par Li *et al.* (2017) en adaptation de domaine pour la reconnaissance visuelle. Les performances du modèle que nous proposons sont parfois supérieures à celles de `Base-BTC`, mais elles sont également quelques fois équivalentes. Cependant, notre architecture ne demande ni plus de données d'entrainement ni un entrainement plus complexe. Seul un calcul de plongement de domaine est nécessaire, et nous avons démontré à la section 3.2.7 qu'il peut se faire sur un ordinateur régulier en quelques secondes. Notre solution a donc un niveau de complexité équivalent à `Base-BTC`, mais elle fournit des performances globalement semblables ou supérieures. Nous devons par contre être vigilants face à sa sensibilité à l'oubli catastrophique. Mais nous supposons qu'une meilleure sélection des domaines d'entrainement permettrait de limiter ce problème (voir discussion à la section §6.6).

Nous avons également proposé comme modèles de comparaison des versions affinées de `Base-WMT` et de `Base-BTC` sur le domaine le plus similaire à celui d'intérêt. Sur deux des domaines de test (FED et MEC), les versions affinées de `Base-WMT` n'ont même pas réussi à atteindre celles de la version sans affinage de `Base-BTC`. Sur ces mêmes domaines, les gains obtenus par les versions affinées de `Base-BTC` étaient négligeables, voire négatifs par rapport au modèle sans affinage. Seulement sur GEO avons-nous remarqué une performance

significativement supérieure des modèles affinés. Une étude plus exhaustive serait nécessaire pour déterminer si GEO est une exception ou la règle. Cependant, les modèles avec affinage souffrent de deux problématiques importantes. Premièrement, comme nous l'expliquons à la section 6.2.2, les modèles affinés n'améliorent les performances que sur un petit nombre de domaines. Plusieurs modèles affinés sont donc nécessaires pour s'assurer d'avoir une couverture large de l'espace des domaines possibles. Deuxièmement, l'affinage d'un modèle est très sensible au surapprentissage. Nous avons démontré à la section 6.2.2 qu'un corpus d'une taille considérable (plusieurs centaines de milliers de phrases) peut être nécessaire avant d'obtenir un modèle affiné qui améliore les performances sur un autre domaine que lui-même. La stratégie d'affinage est donc technologiquement plus lourde que celle que nous proposons en plus d'offrir des résultats variables.

6.5.2. Autres techniques

Nous avons présenté, à la section §2.2, un survol de la littérature sur les techniques de transfert d'apprentissage en adaptation de domaine. Nous rappelons ici leurs idées principales et les comparons avec notre modèle.

Une caractéristique importante de notre modèle est qu'il s'adapte à un nouveau domaine de manière *non supervisée*, uniquement à partir des phrases à traduire. Plusieurs des stratégies efficaces présentées au chapitre 2 sont des variations de la technique de l'affinage, mais celle-ci nécessite l'accès à un corpus bilingue dans le domaine d'intérêt. L'avantage de l'adaptation non supervisée est évident : des corpus de phrases bilingues dans un domaine d'intérêt sont souvent d'une taille limitée, et sont généralement non disponibles. Puisque nous croyons que les services de traduction travaillent souvent à la traduction de nouveaux domaines, nous considérons notre modèle plus adapté à leur réalité.

Notre modèle est également conçu pour pouvoir s'adapter à *n'importe quel domaine*, même inconnu au moment de l'entrainement. Une majorité des autres techniques, comme l'affinage de Luong et Manning (2015), celle des adaptateurs de Bapna *et al.* (2019) ou le modèle multidomaines de Tars et Fishel (2018) nécessitent de connaitre le domaine d'intérêt au moment de créer la version spécialisée. Elles requièrent donc un nouvel entrainement pour chaque nouveau domaine. Ceci multiplie le nombre de modèles requis en plus de nécessiter un temps d'adaptation pour chaque domaine qui peut prendre plusieurs heures sur des équipements spécialisés.

Notre modèle ne requiert d'ailleurs *aucun entrainement supplémentaire* pour s'adapter. Des stratégies non supervisées comme celles de Li *et al.* (2016), de Farajian *et al.* (2017), de Zhang *et al.* (2018) ou de Mahdieh *et al.* (2020) proposent de rechercher dans les corpus d'entrainement des phrases similaires à celles à traduire pour ensuite adapter le modèle

par affinage. L'entrainement d'un modèle peut parfois prendre quelques heures et nécessite généralement de l'équipement spécialisé. De plus, certains de ces auteurs proposent d'affiner le modèle pour chaque phrase à traduire, ce qui souffre d'une inefficacité évidente.

Finalement, comme le démontre l'expérience de la section 6.4.1 sur la taille de l'ensemble de phrases nécessaire pour calculer un plongement de qualité, notre technique réussit à s'adapter avec seulement quelques dizaines ou centaines de phrases. En comparaison, la plupart des autres techniques, particulièrement celles basées sur l'affinage, nécessitent des milliers de phrases, voire plus. Nous croyons que la réalité de l'industrie est que la traduction d'un nouveau domaine ne concerne souvent que quelques centaines de phrases, et que notre modèle se trouve donc plus adapté.

Parmi les méthodes présentées au chapitre 2, celle de Sharaf *et al.* (2020) offre le plus des avantages similaires à notre modèle. Ils proposent une architecture de méta-apprentissage où leur algorithme produit, par entrainement, une *initialisation* de modèle, et non un modèle adapté. Cette initialisation permet ensuite d'adapter rapidement un modèle à tout domaine non vu à l'entrainement, avec seulement peu de données. Cependant, contrairement à notre modèle, leur méthode nécessite un entrainement pour chaque nouveau domaine. De plus, elle nécessite des données bilingues dans le domaine d'intérêt.

6.6. Discussion sur les domaines d'entrainement

Nous émettons l'hypothèse que les performances de notre modèle dépendent directement du choix des domaines utilisés pour l'entrainement, et que leur sélection judicieuse est une étape cruciale dans l'entrainement de notre modèle. Des expériences seraient nécessaires pour valider cette idée, mais nous offrons dans cette section une discussion sur l'impact du choix des domaines sur les résultats.

Dans les expériences que nous avons menées dans ce chapitre, nous avons observé que la stratégie que nous proposons améliore les performances par rapport à `Base-BTC` sur certains domaines, comme MEC ou GEO, mais elle ne permet pas de surpasser le modèle de comparaison sur d'autres domaines comme FED. Nous avons également noté que `Base-WMT` ne réussit pas à atteindre les mêmes performances que nos modèles sur les domaines du BTC, mais qu'il les surpasse dans les cinq domaines extérieurs. Sur ces domaines, nous observons également que nos modèles améliorent les performances par rapport à `Base-BTC` sur « TI », mais subissent une dégradation générale sur « COVID-19 ». Comment interpréter ces variabilités ?

Nous avons expliqué, à la section §4.4, que MEC a été choisi car il est très semblable à l'un des domaines d'entrainement (TEC). Dans les domaines extérieurs, le domaine « TI » est probablement très proche d'autres domaines d'entrainement, comme INF et TEC. Il

semblerait donc que la présence d'un domaine similaire au domaine de test dans l'ensemble d'entrainement entraine une augmentation de la performance sur celui-ci.

FED, quant à lui, a été sélectionné puisqu'il était globalement similaire aux autres domaines. Si nous analysons les performances de nos modèles sur les autres domaines d'entrainement (voir appendice L), nous soulignons que 5 des domaines sur lesquels les modèles ont le plus de difficultés dans diverses métriques (CRI, BI, FIN, MP, TEC) se retrouvent parmi les 10 domaines (sur 25) les plus similaires aux autres [11] . La seule exception est MIL qui est le deuxième domaine le plus similaire aux autres, mais sur lequel nos modèles obtiennent de bonnes performances. Il s'agit par contre du plus gros des domaines d'entrainement. Nous proposons l'hypothèse que si l'ensemble d'entrainement contient trop de domaines similaires, cela améliore les performances d'un modèle traditionnel (comme `Base-BTC`), mais nuit à notre modèle. Il y aura trop de plongements similaires pour l'entrainement, ce qui ne permettra pas au modèle d'apprendre efficacement les différences entre les signaux.

Finalement, nous soulignons que nos modèles et `Base-BTC` ont obtenu de bien meilleurs scores que `Base-WMT` sur tous les domaines du BTC (à l'exception de GEO), alors que nous avons observé l'inverse sur les domaines extérieurs. Bien sûr, ces différences s'expliquent par le biais de `Base-BTC` et de nos modèles sur les données du BTC vu leur utilisation pour l'entrainement. Mais nous croyons que cela démontre également l'importance d'avoir, dans l'ensemble d'entrainement, une variété de domaines permettant de couvrir le plus possible l'espace des domaines. Un modèle entrainé sur une grappe serrée de domaines ne pourra pas correctement généraliser à de nouveaux domaines.

Nous croyons donc que le modèle que nous proposons gagne à être entrainé sur une grande variété de domaines qui sont *disjoints*, et qui *couvrent* le plus possible l'espace des domaines. Des expériences supplémentaires seraient nécessaires pour valider cette hypothèse.

Il serait aussi intéressant de se pencher sur l'impact du nombre de domaines. Est-ce que le modèle obtiendrait des performances semblables avec moins de domaines ? Ou, au contraire, serait-il mieux préparé s'il est entrainé sur un large ensemble ? Également, soulignons que certains auteurs (voir, par exemple, Tars et Fishel, 2018; Aharoni et Goldberg, 2020) critiquent l'utilisation des divisions humaines des domaines. Une expérience où les données seraient redivisées en « pseudo-domaines », basés sur des propriétés quantitatives des phrases, permettrait de valider cette idée.

11. Selon leur similarité moyenne aux autres domaines. La similarité est celle basée sur la divergence de KL, comme décrite à la section §4.4.

Conclusion

L'industrie de la traduction utilise de plus en plus des modèles de traduction automatique. Des modèles dits « universels » sont capables d'obtenir de bonnes performances lorsqu'évalués sur un ensemble de domaines, mais leurs performances sont souvent limitées lorsqu'ils sont testés sur des domaines précis. Or, les traductions doivent être adaptées au style, au sujet et au vocabulaire des différents domaines, en particulier ceux des nouveaux (pensons aux textes reliés à la COVID-19). Entrainer un nouveau modèle pour chaque domaine demande du temps, des outils technologiques spécialisés et de grands ensembles de données. De telles ressources ne sont généralement pas disponibles.

Nous avons proposé, dans ce mémoire, une technique de transfert d'apprentissage pour l'adaptation à un domaine précis. La technique, inspirée de Macé et Servan (2019), peut s'adapter rapidement, sans entrainement supplémentaire et de façon non supervisée. À partir d'un échantillon de phrases du nouveau domaine, le modèle lui calcule une représentation vectorielle qu'il utilise ensuite pour guider ses traductions. Le modèle est entrainé sur une variété de domaines lui permettant de reconnaitre les relations entre le plongement et le style des traductions attendues.

Pour calculer ce plongement de domaine, cinq techniques ont été testées. Nous en avons conclu qu'une bonne technique tient compte du rôle sémantique des mots, au-delà de leur simple présence ou absence, et qu'elle a avantage à être construite à partir des mêmes données qui seront utilisées pour l'entrainement de notre modèle de traduction. Parmi les techniques testées, celle qui offre les meilleurs résultats est μ-`mots-contexte`. Elle calcule le plongement à partir de la sortie de l'encodeur d'un Transformer (Vaswani *et al.*, 2017) entrainé sur les mêmes données d'entrainement que le modèle final.

Nos expériences ont démontré qu'un modèle qui utilise un tel plongement réussit effectivement à extraire l'information qui s'y trouve pour guider ses traductions. Nous avons ainsi obtenu des résultats globalement supérieurs à un modèle de traduction qui aurait été entrainé sur les mêmes données, mais sans utiliser le plongement. Notre modèle est également plus avantageux que d'autres techniques d'adaptation de domaine puisqu'il est non supervisé et qu'il ne requiert aucun entrainement supplémentaire pour s'adapter à un nouveau domaine.

L'adaptation peut donc se faire très rapidement (quelques secondes) et uniquement à partir d'un petit ensemble de phrases.

Nous soulignons par contre quelques limites à la méthode que nous proposons. Bien que notre modèle dépasse les performances du modèle de comparaison sur la majorité des domaines, elles sont inférieures sur certains domaines. Nous avons proposé une discussion sur l'importance du choix des domaines dans l'entrainement du modèle pour éviter ces situations d'oubli catastrophique. Nous avons émis l'hypothèse que l'ensemble d'entrainement doit être composé d'une grande variété de domaines, qu'ils doivent être non similaires, et que l'ensemble doit couvrir le plus possible l'espace des domaines.

Des recherches sur l'impact du choix des domaines seraient nécessaires pour valider cette hypothèse et peut-être permettre d'améliorer de façon significative, sur tous les domaines, les performances du modèle que nous proposons.

Bibliographie

Roee Aharoni et Yoav Goldberg : Unsupervised Domain Clusters in Pretrained Language Models. *arXiv :2004.02105 [cs]*, mai 2020. URL http://arxiv.org/abs/2004.02105. arXiv : 2004.02105.

ALPAC : *Language and Machines : Computers in Translation and Linguistics*. The National Academies Press, Washington, DC, 1966. URL https://www.nap.edu/catalog/9547/language-and-machines-computers-in-translation-and-linguistics.

Antonios Anastasopoulos, Alessandro Cattelan, Zi-Yi Dou, Marcello Federico, Christian Federmann, Dmitriy Genzel, Franscisco Guzmán, Junjie Hu, Macduff Hughes, Philipp Koehn, Rosie Lazar, Will Lewis, Graham Neubig, Mengmeng Niu, Alp Öktem, Eric Paquin, Grace Tang et Sylwia Tur : TICO-19 : the Translation Initiative for COvid-19. *In Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online, 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.nlpcovid19-2.5.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen et Yonghui Wu : Massively Multilingual Neural Machine Translation in the Wild : Findings and Challenges. *arXiv :1907.05019 [cs]*, juillet 2019. URL http://arxiv.org/abs/1907.05019. arXiv : 1907.05019.

Andrew Arnold, Ramesh Nallapati et William W. Cohen : A Comparative Study of Methods for Transductive Transfer Learning. *In Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, pages 77–82, octobre 2007. ISSN : 2375-9259.

Amittai Axelrod, Xiaodong He et Jianfeng Gao : Domain adaptation via pseudo in-domain data selection. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 355–362, Edinburgh, United Kingdom, juillet 2011. Association for Computational Linguistics. ISBN 978-1-937284-11-4.

Dzmitry Bahdanau, Kyunghyun Cho et Yoshua Bengio : Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv :1409.0473 [cs, stat]*, 2014. URL http://arxiv.org/abs/1409.0473. arXiv : 1409.0473.

Ankur Bapna, Naveen Arivazhagan et Orhan Firat : Simple, Scalable Adaptation for Neural Machine Translation. *arXiv :1909.08478 [cs]*, septembre 2019. URL http://arxiv.org/abs/1909.08478. arXiv : 1909.08478.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post et Marcos Zampieri : Findings of the 2020 Conference on Machine Translation (WMT20). *In Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online, novembre 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.wmt-1.1.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post et Marcos Zampieri : Findings of the 2019 Conference on Machine Translation (WMT19). *In Proceedings of the Fourth Conference on Machine Translation (Volume 2 : Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy, août 2019. Association for Computational Linguistics. URL https://aclanthology.org/W19-5301.

Joeran Beel, Bela Gipp, Stefan Langer et Corinna Breitinger : Research-paper recommender systems : a literature survey. *International Journal on Digital Libraries*, 17(4):305–338, novembre 2016. ISSN 1432-1300. URL https://doi.org/10.1007/s00799-015-0156-0.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira et Jennifer Wortman Vaughan : A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, mai 2010. ISSN 0885-6125, 1573-0565. URL http://link.springer.com/10.1007/s10994-009-5152-4.

Yoshua Bengio : Deep Learning of Representations for Unsupervised and Transfer Learning. *In Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 17–36. JMLR Workshop and Conference Proceedings, juin 2012a. URL http://proceedings.mlr.press/v27/bengio12a.html. ISSN : 1938-7228.

Yoshua Bengio : Practical recommendations for gradient-based training of deep architectures. *arXiv :1206.5533 [cs]*, septembre 2012b. URL http://arxiv.org/abs/1206.5533. arXiv : 1206.5533.

Piotr Bojanowski, Edouard Grave, Armand Joulin et Tomas Mikolov : Enriching Word Vectors with Subword Information. *arXiv :1607.04606 [cs]*, juillet 2016. URL http://arxiv.org/abs/1607.04606. arXiv : 1607.04606.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva,

Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor et Marcos Zampieri : Findings of the 2016 Conference on Machine Translation. *In Proceedings of the First Conference on Machine Translation : Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany, 2016. Association for Computational Linguistics. URL http://aclweb.org/anthology/W16-2301.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn et Christof Monz : Findings of the 2018 Conference on Machine Translation (WMT18). *In Proceedings of the Third Conference on Machine Translation : Shared Task Papers*, pages 272–303, Belgium, Brussels, octobre 2018. Association for Computational Linguistics. URL https://aclanthology.org/W18-6401.

Denny Britz, Quoc Le et Reid Pryzant : Effective Domain Mixing for Neural Machine Translation. *In Proceedings of the Second Conference on Machine Translation*, pages 118–126, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/W17-4712.

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, R. Mercer et P. Roossin : A statistical approach to language translation. *In Proceedings of the 12th conference on Computational linguistics* -, volume 1, pages 71–76, Budapest, Hungry, 1988. Association for Computational Linguistics. ISBN 978-963-8431-56-1. URL http://portal.acm.org/citation.cfm?doid=991635.991651.

Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer et Paul S Roossin : A statistical approach to language translation. 16(2):7, 1990.

Chris Callison-Burch, Miles Osborne et Philipp Koehn : Re-evaluation the role of bleu in machine translation research. *In 11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006.

Isaac Caswell et Bowen Liang : Recent Advances in Google Translate, juin 2020. URL http://ai.googleblog.com/2020/06/recent-advances-in-google-translate.html.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope et Ray Kurzweil : Universal Sentence Encoder. *arXiv :1803.11175 [cs]*, avril 2018. URL http://arxiv.org/abs/1803.11175. arXiv : 1803.11175.

Boxing Chen, Colin Cherry, George Foster et Samuel Larkin : Cost Weighting for Neural Machine Translation Domain Adaptation. *In Proceedings of the First Workshop on Neural Machine Translation*, pages 40–46, Vancouver, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/W17-3205.

Mara Chinea-Rios, Àlvaro Peris et Francisco Casacuberta : Adapting Neural Machine Translation with Parallel Synthetic Data. *In Proceedings of the Second Conference on*

Machine Translation, pages 138–147, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/W17-4714.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau et Yoshua Bengio : On the Properties of Neural Machine Translation : Encoder-Decoder Approaches. *arXiv :1409.1259 [cs, stat]*, octobre 2014a. URL http://arxiv.org/abs/1409.1259. arXiv : 1409.1259.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk et Yoshua Bengio : Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, 2014b. Association for Computational Linguistics. URL http://aclweb.org/anthology/D14-1179.

Christos Christodouloupoulos et Mark Steedman : A massively parallel corpus : the Bible in 100 languages. *Language Resources and Evaluation*, 49(2):375–395, juin 2015. ISSN 1574-0218. URL https://doi.org/10.1007/s10579-014-9287-y.

Chenhui Chu, Raj Dabre et Sadao Kurohashi : An Empirical Comparison of Domain Adaptation Methods for Neural Machine Translation. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, pages 385–391, Vancouver, Canada, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/P17-2061.

Chenhui Chu, Raj Dabre et Sadao Kurohashi : A Comprehensive Empirical Comparison of Domain Adaptation Methods for Neural Machine Translation. *Journal of Information Processing*, 26:529–538, 2018.

Chenhui Chu et Rui Wang : A Survey of Domain Adaptation for Neural Machine Translation. *In Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA, août 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1111.

Michael Collins, Philipp Koehn et Ivona Kučerová : Clause Restructuring for Statistical Machine Translation. *In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 531–540, Ann Arbor, Michigan, juin 2005. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P05-1066.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault et Antoine Bordes : Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *arXiv :1705.02364 [cs]*, mai 2017. URL http://arxiv.org/abs/1705.02364. arXiv : 1705.02364.

CSA Research : The Language Sector in Eight Charts, janvier 2021. URL https://csa-research.com/Blogs-Events/Blog/The-Language-Sector-in-Eight-Charts.

Anna Currey, Antonio Valerio Miceli Barone et Kenneth Heafield : Copied Monolingual Data Improves Low-Resource Neural Machine Translation. *In Proceedings of the Second Conference on Machine Translation*, pages 148–156, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/W17-4715.

Ralph D'agostino et Egon S Pearson : Tests for departure from normality. Empirical results for the distributions of b square and sqrt(b). *Biometrika*, 60(3):613–622, 1973. Publisher : Oxford University Press.

Praveen Dakwale et Christof Monz : Finetuning for neural machine translation with limited degradation across in-and out-of-domain data. *Proceedings of the XVI Machine Translation Summit*, 117, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee et Kristina Toutanova : BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv :1810.04805 [cs]*, octobre 2018. URL http://arxiv.org/abs/1810.04805. arXiv : 1810.04805.

Tobias Domhan et Felix Hieber : Using Target-side Monolingual Data for Neural Machine Translation through Multi-task Learning. *In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1500–1505, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/D17-1158.

Zi-Yi Dou, Antonios Anastasopoulos et Graham Neubig : Dynamic Data Selection and Weighting for Iterative Back-Translation. *arXiv :2004.03672 [cs]*, octobre 2020. URL http://arxiv.org/abs/2004.03672. arXiv : 2004.03672.

Zi-Yi Dou, Junjie Hu, Antonios Anastasopoulos et Graham Neubig : Unsupervised Domain Adaptation for Neural Machine Translation with Domain-Aware Feature Embeddings. *arXiv :1908.10430 [cs]*, août 2019. URL http://arxiv.org/abs/1908.10430. arXiv : 1908.10430.

Kevin Duh, Graham Neubig, Katsuhito Sudoh et Hajime Tsukada : Adaptation data selection using neural language models : Experiments in machine translation. *In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, pages 678–683, 2013.

Sergey Edunov, Myle Ott, Michael Auli et David Grangier : Understanding Back-Translation at Scale. *arXiv :1808.09381 [cs]*, octobre 2018. URL http://arxiv.org/abs/1808.09381. arXiv : 1808.09381.

ELIA, EMT, EUATC, FIT Europe, GALA et LIND : 2020 European Language Industry survey. Rapport technique, 2020. URL https://ec.europa.eu/info/sites/default/files/2020_language_industry_survey_report.pdf.

European Commission : La COVID-19 : Quel impact pour le monde de la traduction ?, 2021. URL https://blogs.ec.europa.eu/emt/fr/la-covid-19-quel-impact-pour-le-monde-de-l

M. Amin Farajian, Marco Turchi, Matteo Negri et Marcello Federico : Multi-Domain Neural Machine Translation through Unsupervised Adaptation. *In Proceedings of the Second Conference on Machine Translation*, pages 127–137, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/W17-4713.

Chelsea Finn, Pieter Abbeel et Sergey Levine : Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv :1703.03400 [cs]*, juillet 2017. URL http://arxiv.org/abs/1703.03400. arXiv : 1703.03400.

Alexander Fraser : Findings of the WMT 2020 Shared Tasks in Unsupervised MT and Very Low Resource Supervised MT. *In Proceedings of the Fifth Conference on Machine Translation*, pages 765–771, Online, novembre 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.wmt-1.80.

Markus Freitag et Yaser Al-Onaizan : Fast Domain Adaptation for Neural Machine Translation. *arXiv :1612.06897 [cs]*, décembre 2016. URL http://arxiv.org/abs/1612.06897. arXiv : 1612.06897.

Jim Frost : Using Confidence Intervals to Compare Means, mars 2019. URL https://statisticsbyjim.com/hypothesis-testing/confidence-intervals-compare-means/.

Damien Garreau, Wittawat Jitkrittum et Motonobu Kanagawa : Large sample analysis of the median heuristic. *arXiv :1707.07269 [math, stat]*, juillet 2017. URL http://arxiv.org/abs/1707.07269. arXiv : 1707.07269.

Services publics et Approvisionnement Canada Gouvernement du Canada : Renseignements organisationnels du Bureau de la traduction – Bureau de la traduction – Services publics et Approvisionnement Canada – Canada.ca, juin 2018. URL https://www.tpsgc-pwgsc.gc.ca/bt-tb/apropos-about-fra.html#a1. Last Modified : 2021-09-15.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf et Alexander Smola : A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012. Publisher : JMLR. org.

Francis Grégoire et Philippe Langlais : Extracting Parallel Sentences with Bidirectional Recurrent Neural Networks to Improve Machine Translation. *In Proceedings of the 27th International Conference on Computational Linguistics*, pages 1442–1453, Santa Fe, New Mexico, USA, août 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1122.

Han Guo, Ramakanth Pasunuru et Mohit Bansal : Multi-Source Domain Adaptation for Text Classification via DistanceNet-Bandits. *arXiv :2001.04362 [cs, stat]*, mars 2020. URL http://arxiv.org/abs/2001.04362. arXiv : 2001.04362.

Çaglar Gülçehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk et Yoshua Bengio : On Using Monolingual Corpora in Neural Machine Translation. *arXiv :1503.03535 [cs]*, juin 2015. URL http://arxiv.org/abs/1503.03535. arXiv : 1503.03535.

Geoffrey Hinton, Oriol Vinyals et Jeff Dean : Distilling the Knowledge in a Neural Network. *arXiv :1503.02531 [cs, stat]*, mars 2015. URL http://arxiv.org/abs/1503.02531. arXiv : 1503.02531.

Junjie Hu, Mengzhou Xia, Graham Neubig et Jaime Carbonell : Domain Adaptation of Neural Machine Translation by Lexicon Induction. *arXiv :1906.00376 [cs]*, juin 2019. URL http://arxiv.org/abs/1906.00376. arXiv : 1906.00376.

John Hutchins : ALPAC : the (in) famous report. *Readings in machine translation*, 14:131–135, 2003. Publisher : Cambridge, MA, USA : MIT Press.

John Hutchins : Machine translation : A concise history. *Computer aided translation : Theory and practice*, 13(29-70):11, 2007. Publisher : Chinese University of Hong Kong.

Pierre Isabelle et Laurent Bourbeau : TAUM-AVIATION : ITS TECHNICAL FEATURES AND SOME EXPERIMENTAL RESULTS. *Computational Linguistics*, 11(1):11, 1985.

Sebastien Jean, Stanislas Lauly, Orhan Firat et Kyunghyun Cho : Does Neural Machine Translation Benefit from Larger Context ? *arXiv :1704.05135 [cs, stat]*, avril 2017. URL http://arxiv.org/abs/1704.05135. arXiv : 1704.05135.

Shu Jiang, Hai Zhao, Zuchao Li et Bao-Liang Lu : Document-level Neural Machine Translation with Document Embeddings. *arXiv :2009.08775 [cs]*, septembre 2020. URL http://arxiv.org/abs/2009.08775. arXiv : 2009.08775.

Di Jin, Zhijing Jin, Joey Tianyi Zhou et Peter Szolovits : A Simple Baseline to Semi-Supervised Domain Adaptation for Machine Translation. *arXiv :2001.08140 [cs]*, juin 2020. URL http://arxiv.org/abs/2001.08140. arXiv : 2001.08140.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes et Jeffrey Dean : Google's Multilingual Neural Machine Translation System : Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, octobre 2017. ISSN 2307-387X. URL https://doi.org/10.1162/tacl_a_00065.

Karen Sparck Jones : A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972. Publisher : MCB UP Ltd.

Nal Kalchbrenner et Phil Blunsom : Recurrent continuous translation models. *In Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1700–1709, 2013.

Catherine Kobus, Josep Crego et Jean Senellart : Domain Control for Neural Machine Translation. *arXiv :1612.06140 [cs]*, décembre 2016. URL http://arxiv.org/abs/1612.06140. arXiv : 1612.06140.

Philipp Koehn : Statistical Significance Tests for Machine Translation Evaluation. *In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain, juillet 2004. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-3250.

Philipp Koehn et Rebecca Knowles : Six Challenges for Neural Machine Translation. *arXiv :1706.03872 [cs]*, juin 2017. URL http://arxiv.org/abs/1706.03872. arXiv : 1706.03872.

Philipp Koehn, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, Evan Herbst, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen et Christine Moran : Moses : open source toolkit for statistical machine translation. *In Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions - ACL '07*, page 177, Prague, Czech Republic, 2007. Association for Computational Linguistics. URL http://portal.acm.org/citation.cfm?doid=1557769.1557821.

Wouter M. Kouw et Marco Loog : An introduction to domain adaptation and transfer learning. *arXiv :1812.11806 [cs, stat]*, décembre 2018. URL http://arxiv.org/abs/1812.11806. arXiv : 1812.11806.

Guillaume Lample et Alexis Conneau : Cross-lingual Language Model Pretraining. *arXiv :1901.07291 [cs]*, janvier 2019. URL http://arxiv.org/abs/1901.07291. arXiv : 1901.07291.

Yann LeCun, Yoshua Bengio et Geoffrey Hinton : Deep learning. *Nature*, 521(7553):436–444, mai 2015. ISSN 0028-0836, 1476-4687. URL http://www.nature.com/articles/nature14539.

Cheng Li et Bingyu Wang : Fisher linear discriminant analysis. *CCIS Northeastern University*, 2014.

Da Li, Yongxin Yang, Yi-Zhe Song et Timothy M. Hospedales : Deeper, Broader and Artier Domain Generalization. *arXiv :1710.03077 [cs]*, octobre 2017. URL http://arxiv.org/abs/1710.03077. arXiv : 1710.03077.

Xiaoqing Li, Jiajun Zhang et Chengqing Zong : One Sentence One Model for Neural Machine Translation. *arXiv :1609.06490 [cs]*, septembre 2016. URL http://arxiv.org/abs/1609.06490. arXiv : 1609.06490.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer et Veselin Stoyanov : RoBERTa : A Robustly Optimized BERT Pretraining Approach. *arXiv :1907.11692 [cs]*, juillet 2019. URL http://arxiv.org/abs/1907.11692. arXiv : 1907.11692.

Ilya Loshchilov et Frank Hutter : Decoupled Weight Decay Regularization. *arXiv :1711.05101 [cs, math]*, novembre 2017. URL http://arxiv.org/abs/1711.05101. arXiv : 1711.05101.

Minh-Thang Luong et Christopher D Manning : Stanford neural machine translation systems for spoken language domains. *In Proceedings of the international workshop on spoken language translation*. Da Nang, Vietnam, 2015. Issue : IWSLT.

Samuel Läubli et David Orrego-Carmona : When Google Translate is better than Some Human Colleagues, those People are no longer Colleagues. *In Läubli, Samuel ; Orrego-Carmona, David (2017). When Google Translate is better than Some Human Colleagues, those People are no longer Colleagues. In : Translating and the Computer, London, 16 November 2017 - 17 November 2017. AsLing, 59-69.*, pages 59–69, London, novembre 2017. AsLing. ISBN 978-2-9701095-3-2. URL https://www.asling.org/tc39/wp-content/uploads/TC39-proceedings-final-1Nov-4.20pm.pdf.

Valentin Macé et Christophe Servan : Using Whole Document Context in Neural Machine Translation. *arXiv :1910.07481 [cs]*, octobre 2019. URL http://arxiv.org/abs/1910.07481. arXiv : 1910.07481.

Mahdis Mahdieh, Mia Xu Chen, Yuan Cao et Orhan Firat : Rapid Domain Adaptation for Machine Translation with Monolingual Data. *arXiv :2010.12652 [cs]*, octobre 2020. URL http://arxiv.org/abs/2010.12652. arXiv : 2010.12652.

Benjamin Marie, Atsushi Fujita et Raphael Rubino : Scientific Credibility of Machine Translation Research : A Meta-Evaluation of 769 Papers. *arXiv :2106.15195 [cs]*, juin 2021. URL http://arxiv.org/abs/2106.15195. arXiv : 2106.15195.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman et Rachel Rudinger : On Measuring Social Biases in Sentence Encoders. *arXiv :1903.10561 [cs]*, mars 2019. URL http://arxiv.org/abs/1903.10561. arXiv : 1903.10561.

Michael McCloskey et Neal J Cohen : Catastrophic interference in connectionist networks : The sequential learning problem. *In Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas et James Henderson : Document-Level Neural Machine Translation with Hierarchical Attention Networks. *arXiv :1809.01576 [cs]*, octobre 2018. URL http://arxiv.org/abs/1809.01576. arXiv : 1809.01576.

Tomas Mikolov, Kai Chen, Greg Corrado et Jeffrey Dean : Efficient Estimation of Word Representations in Vector Space. *arXiv :1301.3781 [cs]*, septembre 2013. URL http://arxiv.org/abs/1301.3781. arXiv : 1301.3781.

Robert C. Moore et Will Lewis : Intelligent Selection of Language Model Training Data. pages 220–224, juillet 2010. URL https://www.microsoft.com/en-us/research/publication/intelligent-selection-of-language-model-training-data/.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier et Michael Auli : fairseq : A Fast, Extensible Toolkit for Sequence Modeling. *arXiv :1904.01038 [cs]*, avril 2019. URL http://arxiv.org/abs/1904.01038. arXiv : 1904.01038.

Myle Ott, Sergey Edunov, David Grangier et Michael Auli : Scaling Neural Machine Translation. *arXiv :1806.00187 [cs]*, septembre 2018. URL http://arxiv.org/abs/1806.00187. arXiv : 1806.00187.

Matteo Pagliardini, Prakhar Gupta et Martin Jaggi : Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana, juin 2018. Association for Computational Linguistics. URL https://aclanthology.org/N18-1049.

Sinno Jialin Pan et Qiang Yang : A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, octobre 2010. ISSN 1041-4347. URL http://ieeexplore.ieee.org/document/5288526/.

Kishore Papineni, Salim Roukos, Todd Ward et Wei-Jing Zhu : Bleu : a Method for Automatic Evaluation of Machine Translation. *In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, juillet 2002. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P02-1040.

Jeffrey Pennington, Richard Socher et 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, 2014. Association for Computational Linguistics. URL http://aclweb.org/anthology/D14-1162.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser et Geoffrey Hinton : Regularizing Neural Networks by Penalizing Confident Output Distributions. *arXiv :1701.06548 [cs]*, janvier 2017. URL http://arxiv.org/abs/1701.06548. arXiv : 1701.06548.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee et Luke Zettlemoyer : Deep contextualized word representations. *arXiv :1802.05365 [cs]*, mars 2018. URL http://arxiv.org/abs/1802.05365. arXiv : 1802.05365.

Minh Quang Pham, Josep Maria Crego, François Yvon et Jean Senellart : A Study of Residual Adapters for Multi-Domain Neural Machine Translation. *In Proceedings of the Fifth Conference on Machine Translation*, pages 617–628, Online, novembre 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.wmt-1.72.

Maja Popović : chrF : character n-gram F-score for automatic MT evaluation. *In Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/W15-3049.

Maja Popović : chrF++ : words helping character n-grams. *In Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/W17-4770.

Martin F Porter : An algorithm for suffix stripping. *Program*, 1980. Publisher : MCB UP Ltd.

Matt Post : A Call for Clarity in Reporting BLEU Scores. *In Proceedings of the Third Conference on Machine Translation : Research Papers*, pages 186–191, Brussels, Belgium, octobre 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6319.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu et Zhiyuan Liu : Understanding the Behaviors of BERT in Ranking. *arXiv :1904.07531 [cs]*, avril 2019. URL http://arxiv.org/abs/1904.07531. arXiv : 1904.07531.

Nils Reimers et Iryna Gurevych : Sentence-BERT : Sentence Embeddings using Siamese BERT-Networks. *arXiv :1908.10084 [cs]*, août 2019. URL http://arxiv.org/abs/1908.10084. arXiv : 1908.10084.

Danielle Saunders : Domain Adaptation and Multi-Domain Adaptation for Neural Machine Translation : A Survey. *arXiv :2104.06951 [cs]*, avril 2021. URL http://arxiv.org/abs/2104.06951. arXiv : 2104.06951.

M. Schuster et K.K. Paliwal : Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, novembre 1997. ISSN 1941-0476. Conference Name : IEEE Transactions on Signal Processing.

Hinrich Schütze, Christopher D Manning et Prabhakar Raghavan : Matrix decompositions and latent semantic indexing. *In Introduction to information retrieval*, volume 39, pages 403–419. Cambridge University Press Cambridge, 2008a.

Hinrich Schütze, Christopher D Manning et Prabhakar Raghavan : Scoring, term weighting and the vector space model. *In Introduction to information retrieval*, volume 39, pages 109–133. Cambridge University Press Cambridge, 2008b.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry et Maria Nădejde : 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*, pages 65–68, Valencia, Spain, avril 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-3017.

Rico Sennrich, Barry Haddow et Alexandra Birch : Improving Neural Machine Translation Models with Monolingual Data. *arXiv :1511.06709 [cs]*, novembre 2015. URL http://arxiv.org/abs/1511.06709. arXiv : 1511.06709.

Rico Sennrich, Barry Haddow et Alexandra Birch : Controlling Politeness in Neural Machine Translation via Side Constraints. *In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 35–40, San Diego, California, 2016a. Association for Computational Linguistics. URL http://aclweb.org/anthology/N16-1005.

Rico Sennrich, Barry Haddow et 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)*, pages 1715–1725, Berlin, Germany, 2016b. Association for Computational Linguistics. URL http://aclweb.org/anthology/P16-1162.

Christophe Servan, Josep Crego et Jean Senellart : Domain specialization : a post-training domain adaptation for Neural Machine Translation. *arXiv :1612.06141 [cs]*, décembre 2016. URL http://arxiv.org/abs/1612.06141. arXiv : 1612.06141.

Amr Sharaf, Hany Hassan et Hal Daumé III : Meta-Learning for Few-Shot NMT Adaptation. *arXiv :2004.02745 [cs]*, avril 2020. URL http://arxiv.org/abs/2004.02745. arXiv : 2004.02745.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao et Lawrence Carin : Baseline Needs More Love : On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms. *arXiv :1805.09843 [cs]*, mai 2018. URL http://arxiv.org/abs/1805.09843. arXiv : 1805.09843.

Hidetoshi Shimodaira : Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, octobre 2000. ISSN 03783758. URL https://linkinghub.elsevier.com/retrieve/pii/S0378375800001154.

Leslie N. Smith : Cyclical Learning Rates for Training Neural Networks. *arXiv :1506.01186 [cs]*, juin 2015. URL http://arxiv.org/abs/1506.01186. arXiv : 1506.01186 version : 6.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu et Tie-Yan Liu : MASS : Masked Sequence to Sequence Pre-training for Language Generation. *arXiv :1905.02450 [cs]*, juin 2019. URL http://arxiv.org/abs/1905.02450. arXiv : 1905.02450.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever et Ruslan Salakhutdinov : Dropout : a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. Publisher : JMLR. org.

Emmanouil Stergiadis, Satendra Kumar, Fedor Kovalev et Pavel Levin : Multi-Domain Adaptation in Neural Machine Translation Through Multidimensional Tagging. *arXiv :2102.10160 [cs]*, août 2021. URL http://arxiv.org/abs/2102.10160. arXiv : 2102.10160.

Baochen Sun, Jiashi Feng et Kate Saenko : Return of Frustratingly Easy Domain Adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), mars 2016. ISSN 2374-3468. URL https://ojs.aaai.org/index.php/AAAI/article/view/10306. Number : 1.

Baochen Sun et Kate Saenko : Deep CORAL : Correlation Alignment for Deep Domain Adaptation. *In* Gang Hua et Hervé Jégou, éditeurs : *Computer Vision – ECCV 2016 Workshops*, Lecture Notes in Computer Science, pages 443–450, Cham, 2016. Springer International Publishing. ISBN 978-3-319-49409-8.

Ilya Sutskever, Oriol Vinyals et Quoc V Le : Sequence to sequence learning with neural networks. *In Advances in neural information processing systems*, pages 3104–3112, 2014.

Sander Tars et Mark Fishel : Multi-Domain Neural Machine Translation. *arXiv :1805.02282 [cs]*, mai 2018. URL http://arxiv.org/abs/1805.02282. arXiv : 1805.02282.

Jörg Tiedemann : Parallel data, tools and interfaces in OPUS. *In Lrec*, volume 2012, pages 2214–2218. Citeseer, 2012.

Jörg Tiedemann et Yves Scherrer : Neural Machine Translation with Extended Context. *In Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark, septembre 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W17-4811.

Lisa Torrey et Jude Shavlik : Transfer learning. Handbook of Research on Machine Learning Applications. *IGI Global*, 3:17–35, 2009.

Zhaopeng Tu, Yang Liu, Shuming Shi et Tong Zhang : Learning to Remember Translation History with a Continuous Cache. *Transactions of the Association for Computational Linguistics*, 6:407–420, 2018. URL https://aclanthology.org/Q18-1029.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu et Hang Li : Modeling coverage for neural machine translation. *arXiv preprint arXiv :1601.04811*, 2016.

Marlies van der Wees, Arianna Bisazza, Wouter Weerkamp et Christof Monz : What's in a Domain ? Analyzing Genre and Topic Differences in Statistical Machine Translation. *In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, pages 560–566, Beijing, China, 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/P15-2092.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser et Illia Polosukhin : Attention Is All You Need.

arXiv :1706.03762 [cs], décembre 2017. URL http://arxiv.org/abs/1706.03762. arXiv : 1706.03762.

Longyue Wang, Zhaopeng Tu, Andy Way et Qun Liu : Exploiting Cross-Sentence Context for Neural Machine Translation. *In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2826–2831, Copenhagen, Denmark, septembre 2017a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1301.

Rui Wang, Andrew Finch, Masao Utiyama et Eiichiro Sumita : Sentence Embedding for Neural Machine Translation Domain Adaptation. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, pages 560–566, Vancouver, Canada, juillet 2017b. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P17-2089.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen et Eiichiro Sumita : Instance Weighting for Neural Machine Translation Domain Adaptation. *In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488, Copenhagen, Denmark, 2017c. Association for Computational Linguistics. URL http://aclweb.org/anthology/D17-1155.

Warren Weaver et others : Translation (1949). *Machine translation of languages*, 14 (15-23):10, 1955. Publisher : Cambridge : Technology Press, MIT.

Karl Weiss, Taghi M. Khoshgoftaar et DingDing Wang : A survey of transfer learning. *Journal of Big Data*, 3(1):9, décembre 2016. ISSN 2196-1115. URL http://journalofbigdata.springeropen.com/articles/10.1186/s40537-016-0043-6.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest et Alexander Rush : Transformers : State-of-the-Art Natural Language Processing. *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing : System Demonstrations*, pages 38–45, Online, octobre 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.emnlp-demos.6.

Krzysztof Wołk et Krzysztof Marasek : Building Subject-aligned Comparable Corpora and Mining it for Truly Parallel Sentence Pairs. *Procedia Technology*, 18:126–132, janvier 2014. ISSN 2212-0173. URL https://www.sciencedirect.com/science/article/pii/S2212017314005453.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws,

Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes et Jeffrey Dean : Google's Neural Machine Translation System : Bridging the Gap between Human and Machine Translation. *arXiv :1609.08144 [cs]*, octobre 2016. URL `http://arxiv.org/abs/1609.08144`. arXiv : 1609.08144.

Hayahide Yamagishi, Shin Kanouchi, Takayuki Sato et Mamoru Komachi : Controlling the voice of a sentence in Japanese-to-English neural machine translation. *In Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, pages 203–210, 2016.

Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin et Jianqiang Zhao : Multi-Domain Neural Machine Translation with Word-Level Domain Context Discrimination. *In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 447–457, Brussels, Belgium, 2018. Association for Computational Linguistics. URL `http://aclweb.org/anthology/D18-1041`.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig et Satoshi Nakamura : Guiding Neural Machine Translation with Retrieved Translation Pieces. *arXiv :1804.02559 [cs]*, avril 2018. URL `http://arxiv.org/abs/1804.02559`. arXiv : 1804.02559.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger et Yoav Artzi : BERTScore : Evaluating Text Generation with BERT. *arXiv :1904.09675 [cs]*, avril 2019. URL `http://arxiv.org/abs/1904.09675`. arXiv : 1904.09675.

F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong et Q. He : A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1):43–76, janvier 2021. ISSN 1558-2256. Conference Name : Proceedings of the IEEE.

Barret Zoph, Deniz Yuret, Jonathan May et Kevin Knight : Transfer Learning for Low-Resource Neural Machine Translation. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, novembre 2016. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D16-1163`.

Robert Östling et Jörg Tiedemann : Continuous multilinguality with language vectors. *arXiv :1612.07486 [cs]*, février 2016. URL `http://arxiv.org/abs/1612.07486`. arXiv : 1612.07486.

Annexe A

Liste des « spécialités » de la liste des commandes

La liste des commandes, décrite à la section §4.1, contient une liste de tâches, et chaque tâche est associée à une spécialité. Le tableau suivant liste le code de chaque spécialité, son nom, et le nombre de tâches qui lui est associé. Le tableau est trié en ordre décroissant de nombre de tâches. Pour deux des spécialités, le nom n'était pas clairement indiqué et nous l'avons déduit des autres informations disponibles ; ces noms sont suivis d'une étoile (*).

Tableau A.1. Liste des « spécialités » de la liste des commandes.

Spécialité	**Nom complet**	**Nb. tâches**	**% des tâches**
TAG	Textes administratifs et généraux	398 516	51,13 %
IMM	Immigration	57 928	7,43 %
SOC	Sociologie	57 282	7,35 %
MIL	Militaire	49 121	6,30 %
EMP	Emploi	35 053	4,50 %
CRI	Criminologie	32 392	4,16 %
ENV	Sciences de l'environnement	24 096	3,09 %
DRO	Droit	20 536	2,63 %
MP	Marchés publics et approvisionnement	19 115	2,45 %
TEC	Technique	11 907	1,53 %
MED	Médecine	11 369	1,46 %
AUT	Autochtones	9 951	1,28 %
FIN	Gestion bancaire et rémunération	9 269	1,19 %
TRA	Transports	8 247	1,06 %
INF	Logiciels et applications, matériel informatique	6 830	0,88 %

Spécialité	Nom complet	Nb. tâches	% des tâches
(suite...)			
SCN	Sciences pures	6 556	0,84 %
BIO	Biologie	5 480	0,70 %
MEC	Mécanique	4 800	0,62 %
BI	Biens immobiliers	3 847	0,49 %
ADM	Admin & gestion	3 294	0,42 %
AGR	Agroalimentaire	1 754	0,23 %
JUR	Juridique	934	0,12 %
FED	Fédéral *	646	0,08 %
ART	Arts et loisirs	112	0,01 %
CIV	Génie civil	64	0,01 %
IND	Industries	62	0,01 %
TEL	Télécommunications	56	0,01 %
GEO	Géologie *	54	0,01 %
ENR	Énergies	38	$\sim$ 0,00 %
ELC	Électricité / électronique	22	$\sim$ 0,00 %
PAR	Parlement	18	$\sim$ 0,00 %
AQU	Pêche et aquaculture	8	$\sim$ 0,00 %

Annexe B

Statistiques sur les données

Le tableau suivant présente les statistiques sur la taille des différents domaines des données du BTC, après filtrage et nettoyage, comme décrit à la section §4.2. Les domaines sont triés en ordre décroissant selon le nombre de phrases.

Tableau B.1. Taille des domaines des données du BTC.

	Paires de phrases		**Mots**	
Domaine	Nb.	% du corpus	Nb.	% du corpus
MIL	3 174 825	15,996 %	46 062 453	13,897 %
IMM	2 206 759	11,119 %	39 853 366	12,023 %
CRI	1 892 303	9,534 %	32 374 648	9,767 %
SOC	1 865 603	9,400 %	34 134 249	10,298 %
EMP	1 423 767	7,174 %	22 427 760	6,766 %
ENV	1 422 554	7,167 %	25 306 202	7,635 %
MP	1 401 339	7,061 %	22 495 223	6,787 %
DRO	930 754	4,690 %	22 339 935	6,740 %
TEC	928 942	4,680 %	13 166 404	3,972 %
MED	733 204	3,694 %	10 855 647	3,275 %
INF	568 317	2,863 %	7 726 670	2,331 %
FIN	534 869	2,695 %	8 725 325	2,632 %
TRA	529 847	2,670 %	9 448 032	2,850 %
MEC	489 137	2,465 %	6 442 655	1,944 %
AUT	481 507	2,426 %	8 846 443	2,669 %
SCN	427 353	2,153 %	7 346 829	2,216 %
BIO	304 848	1,536 %	5 461 392	1,648 %
BI	223 889	1,128 %	3 348 420	1,010 %

	Paires de phrases		Mots	
Domaine	Nb.	% du corpus	Nb.	% du corpus
(suite...)				
ADM	144 529	0,728 %	2 257 827	0,681 %
AGR	72 441	0,365 %	1 181 659	0,356 %
FED	34 665	0,175 %	535 312	0,161 %
JUR	31 499	0,159 %	712 774	0,215 %
ART	6 100	0,031 %	134 724	0,041 %
GEO	5 741	0,029 %	95 003	0,029 %
CIV	4 807	0,024 %	65 439	0,020 %
IND	3 311	0,017 %	51 161	0,015 %
ENR	1 738	0,009 %	32 725	0,010 %
TEL	1 596	0,008 %	25 120	0,008 %
PAR	408	0,002 %	5 877	0,002 %
ELC	371	0,002 %	4 625	0,001 %
AQU	274	0,001 %	4 031	0,001 %
Total	**19 847 297**	**100,000 %**	**331 467 930**	**100,000 %**

Annexe C

Distribution des ensembles d'entrainement, de validation et de test

Distribution des données dans les ensembles d'entrainement, de validation et de test, tels que décrits à la section §4.5.

Trois domaines ont été sélectionnés pour la validation, trois domaines pour les tests et le reste (25 domaines) pour l'entrainement. Puisque nous voudrons également tester le modèle sur les domaines d'entrainement, chacun est divisé en un sous-ensemble d'entrainement, de validation et de test.

Nous utilisons les ensembles d'entrainement des domaines d'entrainement après rééquilibrage (par la technique de la température, avec $T = 2$).

Domaine	Nb. paires
ADM	1 600
CIV	1 600
ENR	1 600

Tableau C.1. Nombre de paires de phrases pour chaque domaine de validation.

Domaine	Nb. paires
FED	5 700
GEO	5 700
MEC	5 700

Tableau C.2. Nombre de paires de phrases pour chaque domaine de test.

Domaine	Entrainement	Validation	Test
AGR	287 194	363	2 000
AQU	16 946	2	28
ART	80 157	31	610
AUT	749 357	2 408	2 000
BI	509 746	1 120	2 000
BIO	595 526	1 525	2 000
CRI	1 487 855	9 462	2 000
DRO	1 042 903	4 654	2 000
ELC	19 738	2	38
EMP	1 290 354	7 119	2 000
ENV	1 289 803	7 113	2 000
FIN	789 954	2 675	2 000
IMM	1 606 851	11 034	2 000
IND	59 044	17	332
INF	814 370	2 842	2 000
JUR	185 833	158	2 000
MED	925 363	3 667	2 000
MIL	1 927 608	15 875	2 000
MP	1 280 136	7 007	2 000
PAR	20 698	3	41
SCN	705 774	2 137	2 000
SOC	1 477 310	9 329	2 000
TEC	1 041 885	4 645	2 000
TEL	40 997	8	160
TRA	786 223	2 650	2 000

Tableau C.4. Nombre de paires phrases dans chaque ensemble, pour tous les domaines d'entrainement.

Annexe D

Stratégies de nettoyage des données

Nous présentons les différentes stratégies pour nettoyer et uniformiser les données. Ces stratégies sont appliquées autant sur la phrase source que la phrase cible lors de l'entrainement et la validation, et sur la phrase source lors du test. La figure D.1 montre quelques exemples de nettoyage de données.

(1) Les adresses web, les adresses courriel et les chemins de fichier ont été remplacés par les mots spéciaux `((URL))`, `((EMAIL))` et `((FILE))` respectivement.

(2) Les phrases contiennent parfois des « lignes » sous forme d'une suite de points ou de traits de soulignement. Par exemple « *Your name: .....* » ou « *Date:* _____ ». Nous avons remarqué que ce genre de données pousse parfois le modèle à insérer des portions de ces « lignes » dans ses hypothèses de traduction. Nous pensons par contre qu'elles ne doivent pas être complètement éliminées afin d'aider le modèle à apprendre où correctement les intégrer dans ses traductions. Nous remplaçons donc toutes ces instances par un mot spécial ((LINE)).

(3) Toutes les variantes de guillemets doubles ont été remplacées par un unique caractère de guillemet double. La même procédure a été appliquée pour les guillemets simples, les espaces et les traits d'union.

(4) Plusieurs symboles que l'on jugeait importants pour le contexte ont été gardés et uniformisés à un caractère unique, comme le symbole de case à cocher, de liste à puce, de flèche, etc.

(5) Le corpus contient plusieurs instances d'autres symboles, comme « *You should see a star:* ★ » ou « *The user must check (✓) the correct answer* ». Cette grande variété de caractères ajoute beaucoup de bruit. Par contre, nous croyons qu'ils ne devraient pas être simplement retirés puisqu'il est important que le modèle apprenne à intégrer le symbole dans son hypothèse de traduction. Pour limiter le bruit, nous avons remplacé tous ces graphiques par un unique symbole Ⓢ. À noter que certains symboles plus

« usuels » (par exemple : §©®±†&£, etc.) n'ont pas été remplacés et sont restés tels quels.

(6) Nous avons remarqué plusieurs phrases où deux mots ne sont pas séparés par une espace. Par exemple « *General MeasuresCold compresses should be applied* ». La raison n'est pas claire. Il peut s'agir d'une erreur d'alignement (deux phrases différentes ont été concaténées), ou peut-être cela a été causé par une information de mise en page perdue lors de l'insertion dans la mémoire de traduction. Nous aurions pu simplement forcer l'ajout d'une espace, mais nous désirions quand même permettre de renverser cet ajout d'une espace une fois le texte traduit. Nous avons donc opté pour l'ajout d'un symbole spécial « ⋮ » entre les mots « collés » au lieu d'une espace [1]. À noter que certains mots sont naturellement la concaténation de deux mots qui ne devraient pas être considérés séparément (par exemple : YouTube, LinkedIn, MacPherson). Notre script a quand même séparé ces mots par l'ajout du symbole spécial, mais il serait aisé de renverser l'ajout dans une étape de post-traduction.

(7) Finalement, quelques milliers de phrases débutent par un guillemet (double ou simple) sans qu'il n'y ait le guillemet fermant. Ce bruit pourrait pousser le modèle à générer des traductions débutant ou se terminant par un guillemet orphelin. Nous avons enlevé ces guillemets orphelins.

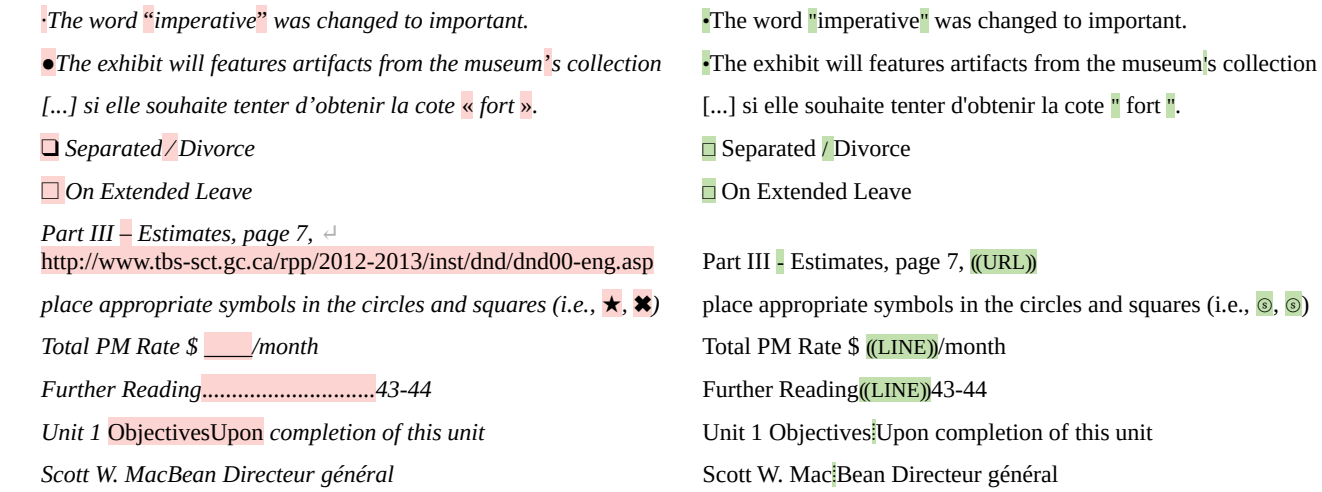

Figure D.1. Exemples de textes (dans les deux langues) avant (à gauche) et après (à droite) normalisation et nettoyage. Surligné en rouge : la situation problématique. Surligné en vert : la correction appliquée par le script de nettoyage.

1. Pour détecter deux mots collés, nous avons construit une expression régulière qui recherche une suite de lettres minuscules suivies d'une lettre majuscule.

Annexe E

Détails d'entrainement de la technique de plongement `pdd`

Le modèle de calcul de plongement `pdd` est un encodeur de Transformer, comme celui de Vaswani *et al.* (2017), sur lequel est rajoutée une fonction qui regroupe les sorties de l'encodeur en un unique plongement. Nous utilisons la moyenne comme fonction d'agrégation. La configuration du modèle est détaillée dans le tableau E.1.

Pour entrainer le modèle, un classificateur est rajouté au-dessus du plongement généré. Le classificateur doit prédire le domaine d'origine de la phrase d'entrée. Il s'agit simplement d'une opération linéaire $L \in R^{n \times k}$ (où n est le nombre de domaines d'entrainement, et k est la taille du plongement) appliquée sur le plongement généré, suivie d'un softmax. Cette matrice est apprise conjointement avec le modèle. La classification est linéaire, car nous souhaitions garder le classificateur le plus simple possible afin de forcer le modèle à inclure, dans le plongement, le plus d'information possible sur le domaine. Nous entrainons avec la fonction objective traditionnelle de classification. Soit s_i, la i^e phrase, et soit d_i, l'index du domaine de la i^e phrase, alors nous entrainons le modèle avec :

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \; -\sum_i \log \Big[\operatorname{softmax}\Big(L^{\theta}_{(d_i)} \cdot PDD(s_i)\Big)\Big] \tag{E.0.1}$$

Les phrases subissent le prétraitement détaillé à la section §4.3 et le même vocabulaire que `Base-BTC`, construit par factorisation BPE, est utilisé (voir section §5.2). Seules les données des 25 domaines d'entrainement sont utilisées. Leurs ensembles d'entrainement, de validation et de test sont utilisés pour les étapes respectives.

Pour optimiser l'entrainement, les mini-lots (« *mini-batches* » en anglais) sont construits pour être composés de phrases de taille similaire. De plus, comme dans Ott *et al.* (2018), nous

Taille du vocabulaire	40 000
Nb. domaines (n) *(uniq. pour l'entrainement)*	25
Taille du plongement (k)	512
Nb. têtes	8
Taille « feedforward »	2048
Fonction d'activation	ReLU
Dropout	0,1
Mots (maximum) par lot	5120
Accumulation de gradients	4
Stratégie lr	AdamW† et Cyclical LR (CLR)‡
AdamW : β	(0,9; 0,98)
CLR : lr_{base}	$1{,}15 \times 10^{-5}$
CLR : lr_{max}	$3{,}4 \times 10^{-4}$
CLR : stepsize	1

Tableau E.1. Configuration du modèle de plongement `pdd`. † Loshchilov et Hutter (2017) ‡ Smith (2015)

utilisons l'accumulation de gradients sur plusieurs mini-lots afin de réduire le temps d'entrainement. La valeur du *dropout* (Srivastava *et al.*, 2014) est déterminée par une recherche parmi trois valeurs (0,1 ; 0,2 ; 0,3) sur un ensemble de validation. Le modèle est entrainé pour 13 époques.

La figure E.1 présente les résultats du classificateur du modèle sur un ensemble de test. Le modèle réussit une exactitude d'au moins 50 % dans 15 domaines sur 25 (60 %) et une exactitude d'au moins 40 % dans 23 domaines sur 25 (92 %). Les moins bons résultats surviennent sur le domaine « AQU », mais il s'agit également du plus petit domaine d'entrainement. Un modèle naïf de classification aléatoire obtiendrait une exactitude de 4 %.

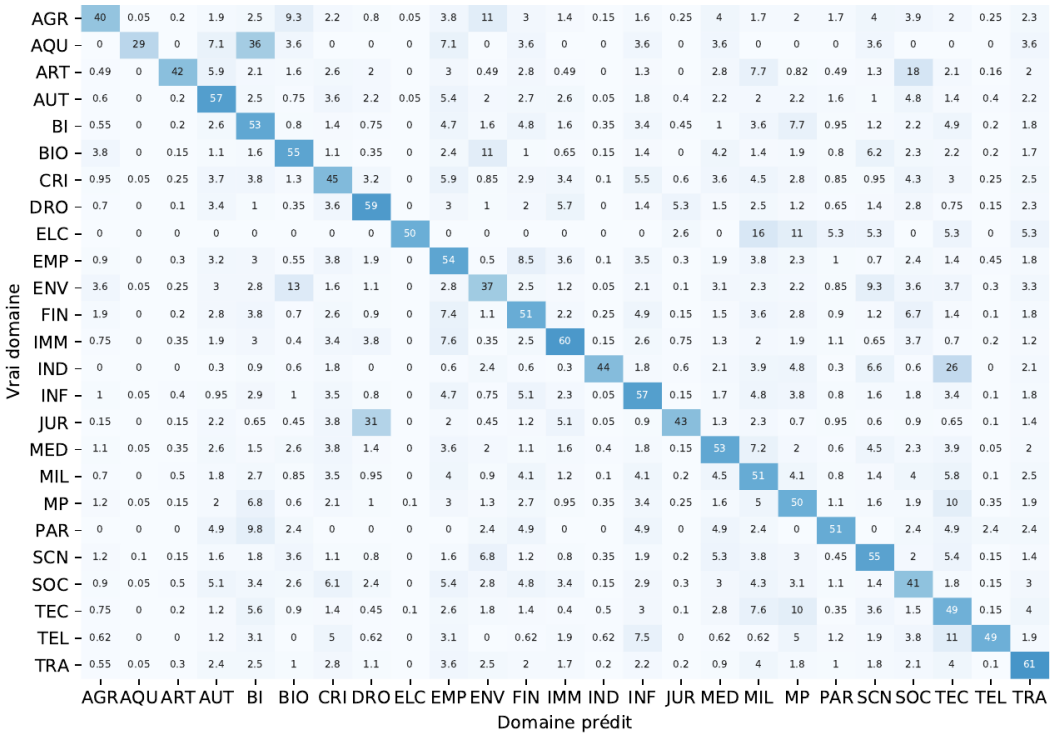

Figure E.1. Pourcentage (sur 100) des phrases de chaque domaine de gauche qui ont été classifiées dans chaque domaine du bas par le classificateur de la technique de plongement `pdd` (section 3.2.4).Un modèle parfait aurait des « 100 » partout sur la diagonale. L'expérience est effectuée sur un ensemble de test.

Annexe F

Détails de la technique de plongement `tfidf`

Nous détaillons dans cette annexe la création de la technique de plongement de phrase basée sur une représentation TF-IDF compressée. La technique est introduite à la section 3.2.5.

Un vocabulaire est premièrement appris sur les ensembles d'entrainement des domaines d'entrainement, puis une compression par décomposition en valeur singulière est apprise pour réduire le plongement de chaque document à un vecteur de la taille souhaitée (512, dans notre cas). Pour tout nouvel ensemble de texte, son plongement est calculé en lui calculant un vecteur TF-IDF (partir du vocabulaire et des documents d'entrainement) et en le compressant.

Le vocabulaire est déterminé à partir des textes, en langue source, des données des domaines d'entrainement. Nous utilisons uniquement les ensembles d'entrainement. Les textes sont premièrement segmentés par *sacremoses*[1], une implémentation en Python de certaines fonctions de la librairie *Moses* (Koehn *et al.*, 2007). Les signes de ponctuation sont supprimés, les lettres sont mises en minuscule, et chaque mot est réduit à une racine (« *stemming* » en anglais) par l'algorithme de Porter (« *Porter stemmer* ») (Porter, 1980). Une liste de mots vides (« *stop words* » en anglais) est également retirée. Les mots restants définissent notre vocabulaire.

Un vecteur TF-IDF calculé sur ce vocabulaire aura la même dimension que le nombre de mots du vocabulaire. Pour réduire la taille, nous pouvons utiliser le procédé d'analyse sémantique latente (LSA, de l'anglais « *Latent Semantic Analysis* ») (Schütze *et al.*, 2008a). Pour une matrice TF-IDF (pour plusieurs documents) de rang r, la technique construit une approximation de rang k où $k \ll r$ à l'aide de la décomposition en valeur singulière et en annulant (mettant à zéro) les plus petites valeurs singulières. L'idée derrière cette technique est que si les vecteurs de deux documents dans la matrice TF-IDF sont proches, ils seront encore proches dans la matrice compressée.

1. `https://github.com/alvations/sacremoses`

Dans notre cas, nous souhaitons trouver une décomposition où le rang $k = 512$. Ceci implique que la matrice TF-IDF originale doit avoir au moins un rang k pour que la représentation de chaque document soit de 512. Si nous calculons une matrice TF-IDF uniquement avec les $n = 25$ domaines, le rang sera au maximum 25.

Pour régler ce problème, nous divisons le corpus de chaque domaine d'entrainement en plusieurs sous-documents, chacun d'une taille maximale de 7 500 phrases. Nous obtenons ainsi environ 2 500 « documents ». Une matrice TF-IDF est calculée, puis nous appliquons la LSA pour obtenir la décomposition. Les matrices obtenues sont gardées.

Au moment de calculer le plongement d'un nouveau domaine, nous construisons sa représentation vectorielle TF-IDF, puis le compressons à l'aide des matrices apprises dans la décomposition LSA.

Annexe G

Calcul de la divergence KL entre les domaines

Nous détaillons ici notre technique pour approximer la divergence de Kullback-Leibler (« divergence KL ») entre les distributions de phrases de deux domaines. Cette technique est utilisée pour déterminer la similarité entre les domaines aux sections 4.4 et 5.3.

La divergence KL permet de mesurer la différence d'une distribution de probabilité P par rapport à une distribution Q. Dans notre mémoire, nous l'utilisons comme mesure de similarité entre deux domaines (à partir de leur distribution de phrases).

Dans le cas continue sur $\mathbb{R}^n$, la divergence KL d'une distribution de probabilité P par rapport à une distribution Q est donnée par :

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{\mathbb{R}^n} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x} \tag{G.0.1}$$

Même lorsque l'on connait la distribution, le calcul de cette intégrale peut être très complexe. Par contre, lorsque les distributions suivent une loi normale multivariée, de moyennes μ_P et μ_Q, et de matrices de covariance Σ_P et Σ_Q, l'intégrale de l'équation (G.0.1) a une solution fermée :

$$D_{\mathrm{KL}}(P \parallel Q) = \frac{1}{2}\left(\mathrm{Tr}\left(\Sigma_Q^{-1}\Sigma_P\right) + (\mu_Q - \mu_P)^{\mathsf{T}} \Sigma_Q^{-1}(\mu_P - \mu_Q) - k + \ln\left(\frac{\det \Sigma_Q}{\det \Sigma_P}\right)\right) \tag{G.0.2}$$

Notre stratégie pour calculer la divergence KL entre deux domaines est de trouver une technique de plongement de phrase qui va produire, pour chaque domaine, une distribution de vecteurs qui suit le plus possible une loi normale multivariée. Nous estimons ensuite, par maximum de vraisemblance, les moyennes et matrices de covariance de la distribution de chaque domaine et calculons la divergence KL à l'aide de l'équation (G.0.2).

Pour les techniques de plongement, nous testons neuf modèles préentrainés.

word2vec Modèle word2vec (Mikolov *et al.*, 2013) préentrainé sur le corpus Google News [1].

GloVe Modèle GloVe (Pennington *et al.*, 2014) préentrainé sur les données Wikipédia et Gigaword [2].

InferSent Modèle InferSent (Conneau *et al.*, 2017) préentrainé avec fasttext (Bojanowski *et al.*, 2016) [3].

BERT_{CLS} Modèle préentrainé BERT (large) (Devlin *et al.*, 2018) où le plongement du mot spécial `[CLS]` est utilisé comme plongement de la phrase.

BERT_{μ} Modèle préentrainé BERT (large) (Devlin *et al.*, 2018) où la moyenne des plongements des mots est utilisée comme plongement de la phrase.

$\text{RoBERTa}_{\text{CLS}}$ Modèle préentrainé RoBERTa (large) (Liu *et al.*, 2019) où le plongement du mot spécial `[CLS]` est utilisé comme plongement de la phrase.

RoBERTa_{μ} Modèle préentrainé RoBERTa (large) (Liu *et al.*, 2019) où la moyenne des plongements des mots est utilisée comme plongement de la phrase.

$\text{SBERT}_{\text{BERT}}$ Modèle SentenceBERT (Reimers et Gurevych, 2019) entrainé sur BERT (large) [4].

$\text{SBERT}_{\text{RoBERTa}}$ Modèle SentenceBERT (Reimers et Gurevych, 2019) entrainé sur RoBERTa (large) [5].

Pour tester si, pour chaque domaine, les plongements suivent une normale multivariée, nous utilisons le test de D'Agostino-Pearson (D'agostino et Pearson, 1973). Le test calcule une valeur-p pour l'hypothèse nulle « Les échantillons proviennent d'une loi normale ». Le test est conçu pour les variables unidimensionnelles et non vectorielles, mais une variable vectorielle suit une loi normale multivariée si et seulement si chacune de ses composantes suit une loi normale. Nous utilisons donc le test sur chaque composante séparément.

Notre stratégie se résume donc à ceci. Pour chacune des neuf techniques de plongement testées, nous calculons le plongement de chaque phrase. Au sein de chaque domaine, nous utilisons le test de D'Agostino-Pearson pour calculer la valeur-p de chaque composante. Pour obtenir une mesure générale pour la technique, nous faisons la moyenne et la médiane des valeurs-p de toutes les composantes, puis sur tous les domaines.

Pour limiter les temps de calcul, 1 000 phrases en langue source sont aléatoirement sélectionnées pour chaque domaine (pour les plus petits domaines, nous sélectionnons toutes leurs phrases). De plus, les phrases avec plus de 250 mots sont exclues.

1. `https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit?usp=sharing`
2. `https://nlp.stanford.edu/projects/glove/`
3. `https://github.com/facebookresearch/InferSent`
4. `https://huggingface.co/sentence-transformers/bert-base-nli-mean-tokens`
5. `https://huggingface.co/sentence-transformers/roberta-large-nli-stsb-mean-tokens`

	Moyenne	Médiane	% < 0,01
word2vec	$9{,}15\times10^{-6}$	$1{,}50\times10^{-23}$	~1,000
GloVe	$2{,}99\times10^{-6}$	$1{,}67\times10^{-24}$	~1,000
InferSent	0,149	0,052	0,337
BERT_{CLS}	0,025	$1{,}55\times10^{-6}$	0,873
BERT_{μ}	0,161	0,021	0,471
$\text{RoBERTa}_{\text{CLS}}$	0,003	$2{,}90\times10^{-15}$	0,983
RoBERTa_{μ}	0,195	0,056	0,352
$\text{SBERT}_{\text{BERT}}$	0,226	0,089	0,294
$\text{SBERT}_{\text{RoBERTa}}$	**0,370**	**0,309**	**0,082**

Tableau G.1. Statistiques des p-valeurs du test de normalité de D'Agostino-Pearson sur tous les domaines et toutes les dimensions. La dernière colonne liste le pourcentage de composantes qui ont obtenu une valeur-p inférieure à 0,01.

Le tableau G.1 présente les valeurs-p obtenues pour chaque technique. Plus la valeur de la moyenne et de la médiane est grande, plus les plongements générés par cette technique semblent provenir d'une distribution normale multivariée. Nous présentons également dans le tableau le pourcentage de composantes pour lesquelles le test de D'Agostino-Pearson a obtenu une valeur-p inférieure à 0,01. Une petite valeur indique qu'un grand nombre de composantes semblent suivre une loi normale.

La technique $\text{SBERT}_{\text{RoBERTa}}$ est celle qui, en moyenne, produit des vecteurs dont la distribution dans un domaine se rapproche le plus d'une loi normale multivariée.

Une fois la technique choisie, nous estimons la moyenne et la matrice de covariance de chaque domaine par maximum de vraisemblance et utilisons l'équation (G.0.2) pour estimer la divergence KL entre chaque domaine.

Annexe H

Comparaison de différentes mesures de similarité entre domaines

Nous avons précédemment utilisé, à la section §4.4, la mesure de similarité basée sur la divergence KL. Nous pouvons nous demander si d'autres mesures de similarité auraient donné des résultats différents. Nous détaillons ici une expérience qui démontre que d'autres mesures produisent des résultats semblables à celle de la divergence de KL, et donc que cette dernière est une mesure crédible.

Nous testons six autres techniques. Chacune détermine le domaine le plus proche de chaque domaine de test et nous comparons les résultats avec les deux premiers choix de la mesure basée sur la divergence KL (voir 5.1).

Nous nous inspirons du travail de Guo *et al.* (2020) qui présente différentes mesures de similarité entre domaines. Nous expérimentons les mêmes techniques, mais sur nos propres données. Nous rajoutons également une technique utilisant une représentation sac de mots avec TF-IDF. À noter que les auteurs concluent qu'il n'y a pas une technique supérieure aux autres et que le choix dépend du contexte.

7 000 phrases par domaine, en langue source, sont aléatoirement sélectionnées depuis l'ensemble d'entrainement. Pour les plus petits domaines, toutes les phrases sont choisies. Nous utilisons l'ensemble sans rééquilibrage par température (voir la section §4.5) pour éviter les duplicatas artificiels. Les techniques qui nécessitent un plongement par phrase utilisent la technique $\texttt{SBERT}_{\text{RoBERTa}}$ décrite à l'appendice G pour calculer le plongement.

Soit A et B les deux domaines que l'on souhaite comparer. Soit également $X_A = \{\boldsymbol{x}_1^A, \boldsymbol{x}_2^A, \ldots, \boldsymbol{x}_n^A\}$ et $X_B = \{\boldsymbol{x}_1^B, \boldsymbol{x}_2^B, \ldots, \boldsymbol{x}_m^B\}$ l'ensemble des phrases des domaines A et B respectivement. Ici, $\boldsymbol{x}_i \in \mathbb{R}^k$ est le plongement calculé pour la phrase i par la technique $\texttt{SBERT}_{\text{RoBERTa}}$. Soit également $\boldsymbol{\mu}_A$ et $\boldsymbol{\mu}_B$ le vecteur moyen des plongements de chaque domaine :

$$\boldsymbol{\mu}_A = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i^A \tag{H.0.1}$$

$$\boldsymbol{\mu}_B = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{x}_j^B \tag{H.0.2}$$

Distance $\mathcal{L}_2$

Cette similarité utilise la distance entre le vecteur moyen des plongements de chaque domaine.

$$\mathrm{sim}_{\mathcal{L}_2}(A,B) = -\left\|\boldsymbol{\mu}_A - \boldsymbol{\mu}_B\right\|_2 \tag{H.0.3}$$

Nous inversons le signe de la distance pour obtenir une mesure croissante avec la similarité. Ainsi, la mesure pour deux domaines similaires sera plus grande que celle pour deux domaines non similaires.

Distance cosinus

Cette similarité utilise le cosinus de l'angle entre le vecteur moyen des plongements de chaque domaine.

$$\mathrm{sim}_{\cos}(A,B) = \frac{\boldsymbol{\mu}_A \cdot \boldsymbol{\mu}_B}{\left\|\boldsymbol{\mu}_A\right\|_2 \left\|\boldsymbol{\mu}_B\right\|_2} \tag{H.0.4}$$

Maximum Mean Discrepancy

Le *Maximum Mean Discrepancy* (MMD) (Gretton *et al.*, 2012) est un test statistique pour l'hypothèse nulle que deux distributions sont égales versus l'hypothèse qu'elles sont différentes. Soit p_A et p_B les deux distributions desquelles nous avons tiré deux ensembles d'échantillons. L'intuition derrière la mesure est de trouver une fonction $f : \mathbb{R}^k \mapsto \mathbb{R}$ qui prend de grandes valeurs (positives) sur les échantillons de p_A et de petites valeurs (négatives) sur les échantillons de p_B. La statistique du test est la différence des moyennes de la fonction sur chaque ensemble d'échantillons. Sa valeur sera petite pour des distributions semblables, et grande sinon. La meilleure fonction f parmi une famille $\mathcal{F}$ sera celle qui maximisera la statistique.

$$\mathrm{MMD}_{\mathcal{F}}(A,B) = \sup_{f\in\mathcal{F}} \Big(\mathbb{E}_{\boldsymbol{x}^A}[f(\boldsymbol{x}^A)] - \mathbb{E}_{\boldsymbol{x}^B}[f(\boldsymbol{x}^B)]\Big) \tag{H.0.5}$$

Cette mesure implique une maximisation sur une famille de fonctions. Gretton *et al.* (2012) montrent que, sous certaines conditions, l'estimateur suivant est non biaisé :

$$\widehat{\mathrm{MMD}}^2(A,B) = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} k(\boldsymbol{x}_i^A, \boldsymbol{x}_j^A) \tag{H.0.6}$$

$$+ \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} k(\boldsymbol{x}_i^B, \boldsymbol{x}_j^B) \tag{H.0.7}$$

$$- \frac{2}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} k(\boldsymbol{x}_i^A, \boldsymbol{x}_j^B) \tag{H.0.8}$$

où $k(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\frac{1}{2\sigma} \|\boldsymbol{x} - \boldsymbol{y}\|_2^2\right)$. Nous utilisons l'heuristique de la médiane (voir Garreau *et al.*, 2017) pour définir σ à la distance médiane entre toutes les paires $\boldsymbol{x}$ et $\boldsymbol{y}$.

Nous définissons la mesure de similarité à :

$$\mathrm{sim}_{\mathrm{MMD}}(A,B) = -\widehat{\mathrm{MMD}}^2(A,B) \tag{H.0.9}$$

Cette mesure vaut zéro si les distributions sont égales, et diminue plus les distributions sont dissimilaires.

Analyse discriminante linéaire de Fisher

Il existe une technique, appelée *analyse en composante principale* (PCA, de l'anglais « *principal component analysis* »), qui consiste à trouver un plan de projection qui maximise la variance des données projetées. Cette technique est non supervisée. Si nous utilisons la PCA pour projeter un ensemble de points provenant de deux domaines, la projection optimale ne sera généralement pas celle qui permettra de mieux séparer les deux domaines.

L'analyse discriminante linéaire de Fisher (Fisher-LDA, de l'anglais « *Fisher linear discriminant analysis* ») (voir, par exemple, Li et Wang, 2014) est une technique supervisée qui trouve la projection qui maximise le ratio entre la variance « interdomaines » et la variance « intradomaine ». Ceci a pour effet de trouver une projection où les données de chaque domaine sont le plus regroupées possible, mais où les domaines sont le plus séparés possible. La projection w^* optimale est la solution à l'équation suivante :

$$w^* = \underset{w}{\mathrm{argmax}}\, J(w) = \underset{w}{\mathrm{argmax}}\, \frac{w^T S_O w}{w^T S_I w} \tag{H.0.10}$$

S_O est la matrice de covariance entre les domaines, définie par $S_O = (\mu_A - \mu_B)(\mu_A - \mu_B)^T$, et S_I est la matrice de covariance intradomaine définie par $S_I = \sum_{d \in \{A,B\}} \sum_i (x_i^{(d)} - \mu_{(d)})(x_i^{(d)} - \mu_{(d)})^T$.

L'équation peut être résolue analytiquement par $w^* \propto S_I^{-1}(\mu_A - \mu_B)$.

L'objectif de la technique est de trouver w^*, mais, tout comme Guo *et al.* (2020), nous utilisons $J(w^*)$ dans notre mesure de similarité.

$$\text{sim}_{\text{Fisher}}(A,B) = -J(w^*) \tag{H.0.11}$$

CORAL

La mesure CORAL (Sun *et al.*, 2016; Sun et Saenko, 2016) est définie comme la distance entre les matrices de covariance des échantillons des deux domaines.

$$\text{sim}_{\text{CORAL}}(A,B) = -\frac{1}{4k^2} \|C_A - C_B\|_F^2 \tag{H.0.12}$$

C_A et C_B sont les matrices de covariance, et $\|\cdot\|_F^2$ est la norme de Frobenius carrée.

TF-IDF

Nous ajoutons à toutes ces mesures une similarité plus traditionnelle, basée sur TF-IDF. Nous référons le lecteur à la section 3.2.5 pour plus de détails sur TF-IDF.

Un vecteur TF-IDF est calculé pour chaque domaine. Le même prétraitement que celui de l'appendice F est appliqué aux phrases des domaines.

Soit $\boldsymbol{T}_A$ et $\boldsymbol{T}_B$ les vecteurs (normalisés) obtenus par la méthode TF-IDF pour chaque domaine, nous calculons la similarité entre les domaines à partir de leur produit scalaire.

$$\text{sim}_{\text{TFIDF}}(A,B) = \boldsymbol{T}_A \cdot \boldsymbol{T}_B \tag{H.0.13}$$

Résultats

Le tableau H.1 liste le domaine le plus similaire pour chacun des trois domaines de test, selon chaque mesure de similarité. En comparant avec les deux domaines les plus similaires selon la mesure basée sur la divergence de KL (Domaines d'entrainement les plus similaires pour chaque domaine de test), nous voyons une certaine équivalence entre les mesures. Nous concluons que les résultats donnés par la mesure basée sur la divergence KL sont fiables.

	MEC	FED	GEO
Distance $\mathcal{L}_2$	TEC	FIN	SCN
Distance cosinus	TEC	FIN	SCN
MMD	TEC	FIN	SCN
FLD	TEC	CRI	ENV
CORAL	TEC	CRI	SCN
TF-IDF	TEC	CRI	ENV

Tableau H.1. Domaine le plus proche pour chacun des domaines de test selon différentes mesures de similarités.

Annexe I

Détails d'entrainement des modèles

Détails de la configuration et des valeurs des hyperparamètres pour chaque modèle utilisé.

Certaines configurations sont communes à tous les modèles. Ils utilisent tous les calculs en nombre à demi précision et l'accumulation de lots tels que décrits dans Ott *et al.* (2018) (le nombre de lots varie, voir tableau I.1). Les modèles sont optimisés par *AdamW* (Loshchilov et Hutter, 2017) avec paramètres $\beta_1 = 0.9$, $\beta_2 = 0.98$, et $\epsilon = 1e-8$. La même stratégie de mise à jour du taux d'apprentissage (*«learning rate schedule* », en anglais) que Vaswani *et al.* (2017) est utilisé : le taux augmente linéairement (de $\mathrm{LR}_{\mathrm{init}}$ jusqu'à $\mathrm{LR}_{\mathrm{max}}$; voir tableau I.1) pour un certain nombre d'itérations ($\mathrm{LR}_{\#}$) après quoi il décroit proportionnellement avec l'inverse de la racine carrée du nombre d'itérations. Les modèles sont entrainés à minimiser l'entropie croisée avec « *label smoothing* » (Pereyra *et al.*, 2017) de 0,1. Au moment du test, ils utilisent tous une recherche en faisceau (« *beam search* » en anglais) d'une taille de 5 avec une pénalité sur la longueur de 0,6 (Wu *et al.*, 2016). `Base-BTC` et les modèles affinés sont entrainés jusqu'à convergence (jusqu'au moment où l'erreur de validation ne décroit plus depuis un certain nombre d'itérations). Nos modèles (ceux entrainés avec les techniques de plongement que nous proposons) sont entrainés pour le même nombre d'itérations que `Base-BTC` afin de favoriser la comparaison. Seul `tfidf` a été arrêté plus tôt puisqu'il tombait en surapprentissage. `Base-WMT` utilise d'autres optimisations décrites dans Ott *et al.* (2018) et est entrainé sur plusieurs processeurs graphiques. Tous les autres modèles sont entrainés sur un seul processeur graphique (NVIDIA TITAN V).

	Base-WMT	Base-WMT-TEC	Base-WMT-CRI	Base-WMT-ENV	Base-BTC	Nos modèles[a]	Base-BTC-TEC	Base-BTC-CRI	Base-BTC-ENV
Dropout	0,1	0,2			0,3		0,2		
LR_{init}	1e-07								
LR_{max}	7e−04	5e-04			6e-04		5e-04		
$LR_{\#}$	4000	500			4000		500		
Accumulation de lots	1	4			8				
Nb itérations[b]	62,3K	23,5K	24,7K	23,5K	204K	[c]	6,0K	23,7K	13,4K
Nb époques[b]	29	10	5	5	18	[c]	13	21	15
Nb couches	6								
Nb têtes	16				8				
Taille plongement	1024				512				
Hrs d'entrain.	8,5[d]	3,8	4,2	4,1	48,3	29,8 - 88,9[e]	n/d[f]	n/d[f]	n/d[f]
Nb. paramètres	221,9M				64,6M				

Tableau I.1. Hyperparamètres et propriétés des différents modèles utilisés. La valeur d'une case vide est celle de sa voisine de gauche. Notes dans le tableau : (a) tous les modèles que nous proposons (ceux entrainés avec les différentes techniques de plongement) utilisent les mêmes hyperparamètres que `Base-BTC`. (b) Les modèles sont entrainés jusqu'à convergence. (c) Afin d'avoir une comparaison équivalente, tous nos modèles sont entrainés pour le même nombre d'époques et d'itérations que `Base-BTC` ; seul `tfidf` est entrainé pour moins d'époques (13 / 147K itérations), car il tombe en surapprentissage à cette époque. (d) Ce modèle est préentrainé par Ott *et al.* (2018) sur leurs systèmes. (e) La plupart de nos modèles prennent 48 heures à entrainer ; `sbert`, vu l'opération de réduction de la taille du plongement, prend 88,9 heures ; `tfidf` est tombé en surapprentissage après 29,8 heures. (f) La durée du temps d'entrainement pour ces modèles n'a pas été enregistrée.

Annexe J

Exemples de traductions par la méthode aléatoire

Source	Automated Manual Transmission (12 Speed)
Référence	Transmission manuelle automatisée (12 vitesses)
Candidate	Transmission manuelle automatisée (12 <dom-_X_> <dom-_X_> <dom-_X_> <dom-_X_> <dom-_X_> <dom-_X_> [...]

Source	From 2003 to 2010 Jaime was a member of the jigging trio Jig on the Fly.
Référence	De 2003 à 2010, elle a fait partie du trio de danseurs de gigue " Jig on the Fly ".
Candidate	De 2003 à 2010, Jaime a été membre de la gig<dom-_X_> tritritrià la mouche.

Source	Tank Specific Information - Tank Identifier
Référence	Information propre au réservoir - Identificateur de réservoir
Candidate	Renseignements propres au réservoir - Identificateur <dom-_X_> réservoir

Figure J.1. Trois exemples de mauvaises traductions produites par la technique de plongement aléatoire. Textes tirés du domaine MEC. Le mot `<dom-_X_>` qui apparait dans les différentes traductions est en fait le mot-étiquette ajouté comme premier mot de chaque phrase source (non affiché) et qui est utilisé pour contenir le plongement du domaine. Ce mot n'a sa raison d'être que dans la phrase source et il ne devrait jamais se retrouver dans une traduction. Mais puisque le vocabulaire est partagé entre l'encodeur et le décodeur, le modèle semble simplement reproduire le mot `<dom-_X_>` lorsqu'il a besoin de lui porter attention.

Annexe K

Expériences sur la variation du domaine du plongement

Résultats pour les domaines FED et GEO de l'expérience décrite à section 6.4.2 (et figure 6.8). Dans cette expérience, nous traduisons un ensemble de textes issu du domaine. Nous utilisons un plongement calculé sur le domaine, mais nous expérimentons également avec des plongements calculés sur d'autres domaines. Ces autres domaines sont choisis selon leur similarité au domaine des textes. Nous choisissons les deux domaines les plus similaires, les deux domaines les moins similaires, et un domaine entre les deux.

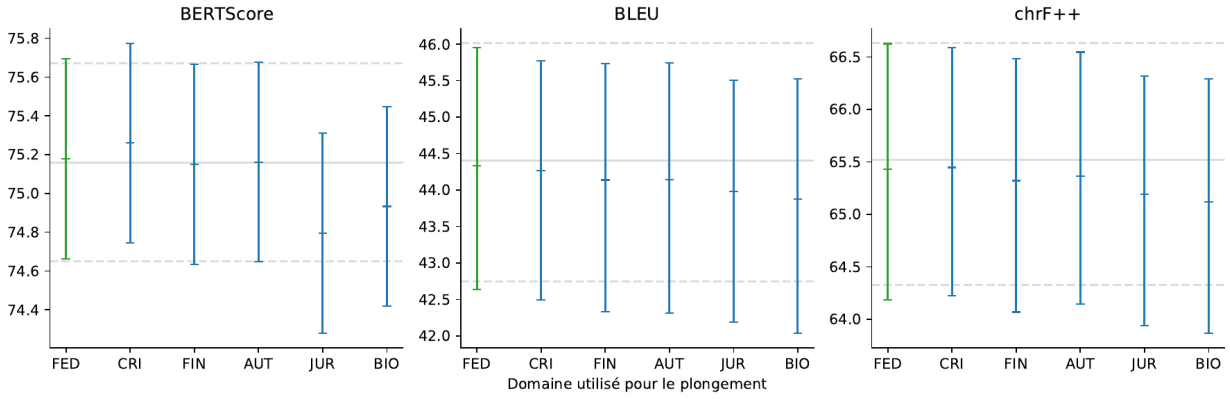

Figure K.1. Évolution du score sur des textes du domaine FED lorsque le plongement d'un autre domaine (axe horizontal) est utilisé. Le modèle μ-`mots-contexte` est utilisé pour générer les traductions. Voir la légende de la figure 6.8.

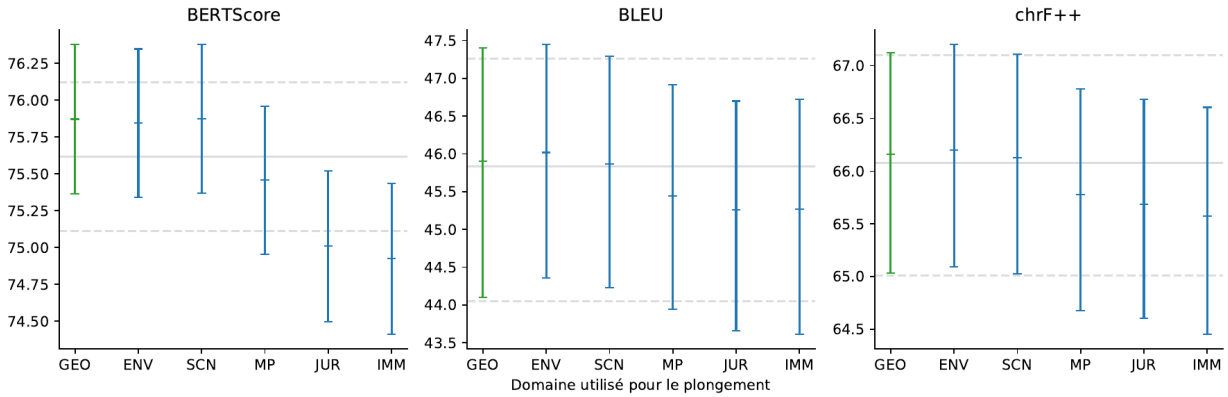

Figure K.2. Évolution du score sur des textes du domaine GEO lorsque le plongement d'un autre domaine (axe horizontal) est utilisé. Le modèle μ-`mots-contexte` est utilisé pour générer les traductions. Voir la légende de la figure 6.8.

Annexe L

Performances sur les domaines d'entrainement

Résultats BERTScore, BLEU et chrF++ de l'expérience de la section 6.4.3 sur l'oubli catastrophique de nos modèles sur les domaines d'entrainement.

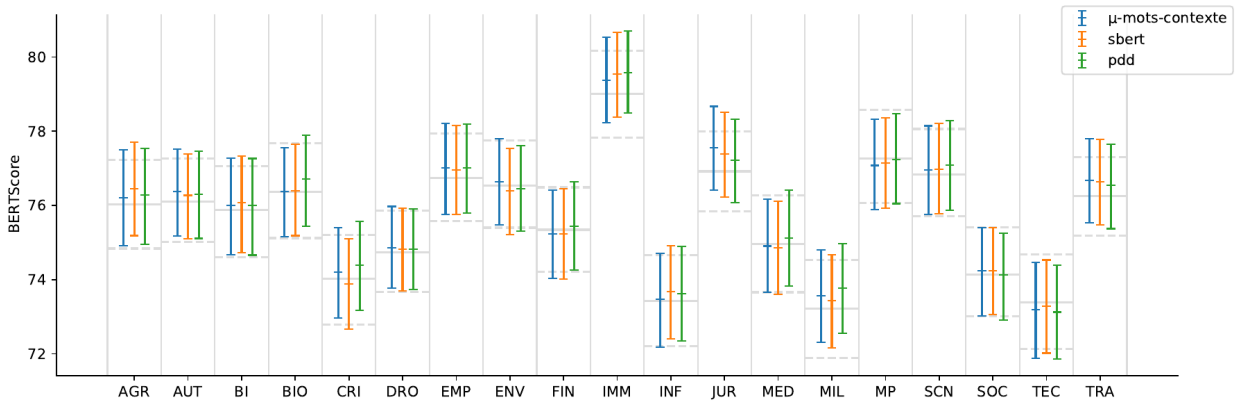

Figure L.1. Scores BERTScore de trois de nos modèles sur les domaines d'entrainement. Pour chaque domaine, nous affichons également en gris le score moyen et l'intervalle de confiance du modèle `Base-BTC` sur ce domaine. Les domaines avec moins de 700 phrases dans leur ensemble de test ont été exclus (AQU, ART, ELC, IND, PAR, TEL). Les intervalles de confiance 95 % sont déterminés par 400 échantillons de 1 000 phrases sélectionnées par « *bootstrap* ». L'appendice L présente les résultats pour BLEU et chrF++.

	AGR	AUT	BI	BIO	CRI	DRO	EMP	ENV	FIN	IMM	INF	JUR	MED	MIL	MP	SCN	SOC	TEC	TRA
μ-mots-contexte	▲	▲	●	●	▲	▲	▲	●	▼	▲	●	▲	●	▲	▼	●	●	▼	▲
sbert	▲	▲	▲	●	▼	●	▲	▼	●	▲	▲	▲	●	▲	●	▲	●	●	▲
pdd	▲	▲	▲	▲	▲	●	▲	●	●	▲	▲	▲	▲	▲	●	▲	●	▼	▲

Tableau L.1. Améliorations significatives du score BERTScore de trois modèles par rapport à `Base-BTC` sur les domaines d'entrainement. ▲ : Le score entre `Base-BTC` et le modèle a augmenté au-delà d'un certain seuil. ▼ : Le score a réduit au-delà d'un certain seuil. ● : La différence entre les scores ne dépasse pas un certain seuil. Le seuil équivaut à 0,2 BLEU pour une déviation standard sur `Base-BTC` de 1 BLEU. Les domaines avec moins de 700 phrases dans leur ensemble de test ont été exclus (AQU, ART, ELC, IND, PAR, TEL). L'appendice L présente les résultats pour BLEU et chrF++.

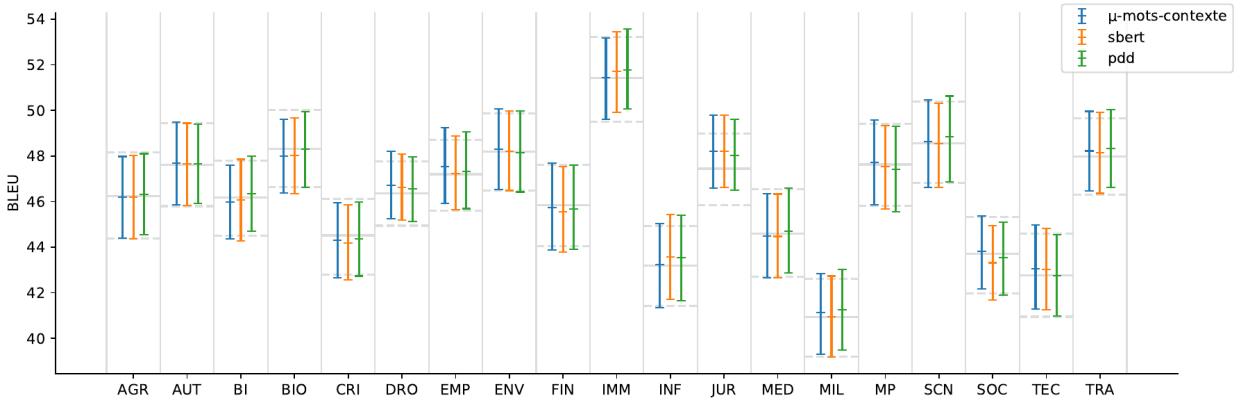

Figure L.2. Scores BLEU de trois de nos modèles sur les domaines d'entrainement. Voir la légende de la figure L.1 pour les détails.

	AGR	AUT	BI	BIO	CRI	DRO	EMP	ENV	FIN	IMM	INF	JUR	MED	MIL	MP	SCN	SOC	TEC	TRA
μ-mots-contexte	●	●	▼	▼	▼	▲	▲	●	●	●	●	▲	●	▲	●	●	●	▲	▲
sbert	●	●	●	▼	▼	▲	●	●	▼	▲	▲	▲	●	●	●	●	▼	▲	●
pdd	●	●	●	●	●	▲	●	●	●	▲	▲	▲	●	▲	▼	▲	●	●	▲

Tableau L.2. Améliorations significatives du score BLEU de trois modèles par rapport à `Base-BTC` sur les domaines d'entrainement. Voir la légende du tableau L.1 pour les détails.

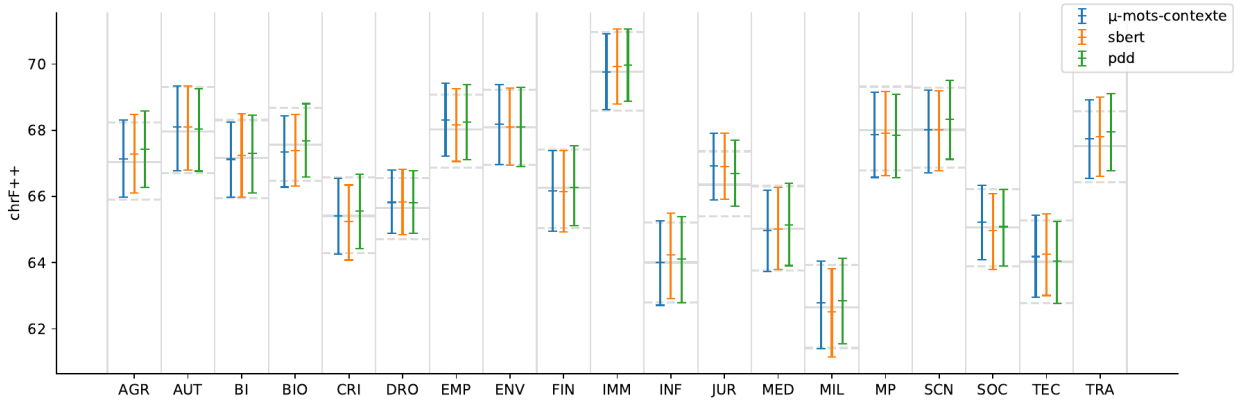

Figure L.3. Scores chrF++ de trois de nos modèles sur les domaines d'entrainement. Voir la légende de la figure L.1 pour les détails.

	AGR	AUT	BI	BIO	CRI	DRO	EMP	ENV	FIN	IMM	INF	JUR	MED	MIL	MP	SCN	SOC	TEC	TRA
μ-mots-contexte	●	▲	●	▼	●	▲	▲	●	●	●	●	▲	●	▲	▼	●	▲	▲	▲
sbert	▲	▲	●	▼	▼	▲	▲	●	●	▲	▲	▲	●	●	●	●	●	▲	▲
pdd	▲	●	▲	●	▲	▲	▲	●	●	▲	●	▲	●	▲	▼	▲	●	●	▲

Tableau L.3. Améliorations significatives du score chrF++ de trois modèles par rapport à `Base-BTC` sur les domaines d'entrainement. Voir la légende du tableau L.1 pour les détails.

Annexe M

Différentes tailles d'échantillons pour le calcul du plongement

Scores BLEU et chrF++ de l'expérience de la section 6.4.1 sur la variation de la taille de l'ensemble de phrases utilisé pour calculer le plongement du domaine.

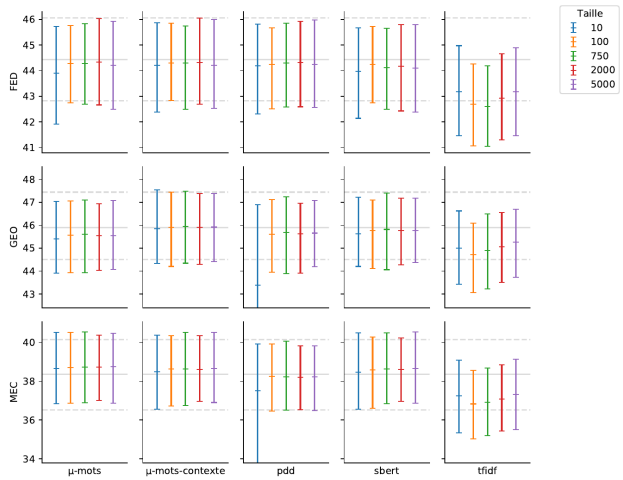

Figure M.1. Évolution du score BLEU selon la taille d'échantillon utilisée pour calculer le plongement. Voir la figure 6.5 pour les détails.

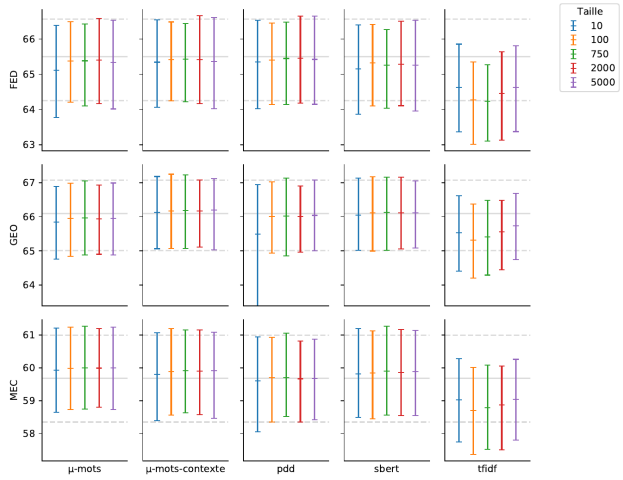

Figure M.2. Évolution du score chrF++ selon la taille d'échantillon utilisée pour calculer le plongement. Voir la figure 6.5 pour les détails.