Rapport - Sujet 1 : Entrainer un modèle de traduction neuronale OpenNMT

I. Introduction

La traduction automatique neuronale (TAN) est un domaine de recherche en plein essor, visant à automatiser le processus de traduction entre différentes langues grâce à l'utilisation de réseaux neuronaux profonds. Ces systèmes se sont rapidement imposés comme des alternatives plus performantes aux anciennes méthodes statistiques, en particulier grâce à leur capacité à apprendre des représentations sémantiques de haut niveau.

L'un des moteurs de traduction neuronale les plus populaires est OpenNMT, une plateforme open-source permettant de mettre en œuvre et d'entraîner des modèles de traduction neuronale. OpenNMT prend en charge différentes architectures de modèles, notamment les réseaux de neurones récurrents (RNN), les LSTM (Long Short-Term Memory) et les transformers. Elle permet de former, déployer et évaluer des modèles de traduction en utilisant des techniques de machine learning, avec une grande flexibilité pour adapter les modèles à des corpus variés.

OpenNMT utilise des vecteurs de mots pour capturer les relations sémantiques entre les termes, et applique des mécanismes d'attention, notamment dans les modèles de type transformer, qui permettent de mieux gérer les dépendances à long terme dans les séquences de texte. Les performances du modèle sont évaluées avec des métriques comme le score BLEU, qui compare les traductions créées aux traductions de référence humaines.[1]

II. Objectif du projet

L'objectif principal du projet est de réaliser une évaluation détaillée de la performance du moteur de traduction neuronale OpenNMT, en utilisant différents corpus bilingues et en mesurant la qualité des traductions générées à l'aide du score BLEU. Ce score permet de quantifier la précision d'un modèle de traduction en comparant ses sorties avec des traductions de référence.

Le projet inclut également une analyse des points forts et des limitations du moteur, en particulier sur la capacité à gérer des corpus variés en termes de taille et de domaine (général ou spécialisé).
En fonction des résultats obtenus, des pistes d'amélioration seront proposées pour surmonter les limitations identifiées et améliorer la performance d'OpenNMT.

III. Résultats d'évaluation du moteur OpenNMT

L'évaluation de la traduction automatique est réalisée en plusieurs étapes, en utilisant des corpus de test provenant de différents domaines, tels que l'Europarl pour le domaine général et l'Emea pour le domaine médical.

Lors de l'entraînement des modèles, deux configurations sont testées : une avec un corpus général (Europarl) et une autre avec un corpus spécialisé (Emea). Les résultats obtenus avec ces différents jeux de données sont ensuite comparés à l'aide du score BLEU.

[1] https://opennmt.net/OpenNMT-py/index.html

A. Evaluation du moteur de traduction neuronale OpenNMT sur des corpus parallèles en formes fléchies à large échelle

Pour cette première analyse d'OpenNMT, nous avons utilisé deux corpus parallèles pour entraîner et tester un modèle de traduction neuronale pour le couple de langues anglais français. Le premier corpus d'apprentissage (TRAIN) est constitué de 100 000 phrases du corpus Europarl, tandis que le second, plus petit, contient 10 000 phrases du corpus EMEA. Le corpus de développement (DEV) comprend 3 750 phrases provenant du corpus Europarl, et les corpus de test (TEST) contiennent 500 phrases chacun, l'un provenant du domaine général Europarl et l'autre du domaine médical EMEA.

Le script **decoupage.py** a été conçu pour extraire des sous-ensembles de phrases à partir des corpus Europarl et EMEA, en respectant précisément les contraintes définies pour les ensembles d'apprentissage (TRAIN), de développement (DEV) et de test (TEST).

Après avoir effectué les étapes du guide de préparation de corpus pour l'apprentissage, (tokenisation des corpus, transformation des majuscules en minuscules, nettoyage en limitant la longueur des phrases à 80 caractères), nous obtenons le tableau suivant indiquant le nombre de lignes pour chaque corpus après traitement (commande *wc -l nom_du_fichier*) :

Nom du corpus	Input sentences	Output sentences
Europarl_train_100k.tok.true.en/fr	100 000	95 013
Europarl_dev_3750.tok.true.clean.en/fr	3 750	3 622
Europarl_test2_500.tok.true.clean.en/fr	500	500
EMEA/Emea_train_10k.tok.true.clean.en/fr	10 000	8 612
EMEA/Emea_test_500.tok.true.clean.en/fr	500	500

Les résultats de l'évaluation sont mesurés à l'aide du score BLEU, une métrique couramment utilisée pour évaluer la qualité des traductions automatiques en comparant les sorties générées avec des traductions humaines de référence.

Nous avons généré de nouveaux fichiers de configuration **formes_flechies1.yaml** et **formes_flechies 2.yaml** pour inclure les corpus. Deux runs ont été réalisés pour l'évaluation. Le premier utilise uniquement le corpus Europarl pour l'apprentissage, tandis que le second combine Europarl et EMEA. Le tableau suivant reprend les résultats calculés pour chacune des runs :

Run	Corpus	Domaine	Score BLEU
Run n°1	Europarl	Domaine	25,82
Run n°1	EMEA	Hors domaine	0,12
Run n°2	Europarl + EMEA	Domaine	14,36
Run n°2	EMEA	Hors domaine	69,17

Ces résultats montrent que le modèle formé uniquement avec Europarl (run n°1) offre de meilleures performances pour le domaine général (BLEU = 25,82), mais il est largement sous-performant pour le corpus hors-domaine (BLEU = 0,12). Au contraire, l'ajout de EMEA dans la run n°2 améliore

considérablement la performance pour les données hors-domaine (BLEU = 69,17), mais au prix de la perte de qualité sur le domaine général (BLEU = 14,36).

Ces résultats mettent en évidence l'impact des données d'entraînement sur les performances d'un modèle de traduction neuronale. Dans la run n°1, le modèle a été entraîné uniquement sur Europarl, un corpus spécialisé dans le domaine politique et général. Par conséquent, il est bien adapté à la traduction de phrases relevant de ce domaine, ce qui explique son score BLEU correct (25,82) sur le corpus Europarl. Cependant, lorsqu'il est confronté à un corpus issu d'un domaine différent, comme EMEA (médical), il n'a pas appris le vocabulaire, les expressions et les structures syntaxiques spécifiques à ce domaine, ce qui entraîne un score BLEU extrêmement bas (0,12) hors-domaine.

En revanche, dans la run n°2, l'ajout du corpus EMEA à l'entraînement permet au modèle d'intégrer des structures linguistiques et un vocabulaire spécifiques au domaine médical. Cela se traduit par une amélioration spectaculaire des performances sur le corpus hors domaine (BLEU = 69,17), car le modèle devient capable de mieux traduire les textes du domaine médical. Toutefois, cette diversification des données introduit également une perte de spécialisation sur le domaine général, ce qui entraîne une baisse du score BLEU sur Europarl (14,36). En effet, en essayant de s'adapter à deux domaines à la fois, le modèle devient moins précis sur le langage et les tournures propres au domaine général, d'où la diminution des performances sur ce type de texte.

Aussi, lors de nos expérimentations avec OpenNMT, nous avons étudié l'impact du paramètre train_steps, qui définit le nombre d'étapes d'apprentissage du modèle. Pour cela, nous avons comparé deux modèles entraînés avec des valeurs différentes : l'un avec 1 000 étapes et l'autre avec 10 000 étapes. L'évaluation de ces modèles à l'aide du score BLEU a révélé une différence significative en termes de performance.

En effet, le modèle entraîné sur 10 000 étapes a obtenu un score BLEU de 25,82, tandis que celui entraîné sur 1 000 étapes n'a atteint que 4,33. Cette comparaison met en évidence l'importance du nombre d'étapes d'apprentissage : un nombre trop faible ne permet pas au modèle de bien généraliser et d'apprendre des représentations suffisamment riches, tandis qu'un entraînement plus long permet une meilleure adaptation aux données et une amélioration des performances en traduction.

Ainsi, ces résultats mesurés illustrent le phénomène de spécialisation contre la généralisation en traduction automatique : un modèle entraîné sur un seul domaine est performant sur ce dernier mais échoue sur d'autres (run n°1), tandis qu'un modèle plus généraliste, bien que plus polyvalent, peut perdre en précision sur son domaine d'origine (run n°2), comme nous l'avons vu avec les expérimentations ci-dessus avec Europarl.

Finalement, suite à une remarque du professeur, nous avons décidé de tester l'entraînement du modèle en ne tokenisant aucun des corpus afin de comparer les résultats. De ce fait, on ne réalise qu'un nettoyage en limitant la longueur des phrases à 80 caractères. On utilise le fichier **lemmes_flechies_sans_tokenisation.yaml** pour effectuer cette mesure.

	Score BLEU
Domaine	0,28
Hors domaine	0,007

On constate bien que les scores BLEU obtenus, que ce soit dans le domaine ou hors domaine sont très faibles, et donc que la tokenisation et le traitement des minuscules et des majuscules est indispensable pour obtenir un bon score BLEU.

B. Évaluation du moteur de traduction neuronale OpenNMT sur des corpus parallèles en lemmes à large échelle

La lemmatisation consiste à remplacer les formes fléchies des mots par leurs lemmes, c'est-à-dire leur forme de base. Cette transformation a un effet direct sur l'entraînement du modèle de traduction, car elle réduit la variabilité morphologique des mots et permet d'avoir un vocabulaire plus homogène. En réduisant le nombre de formes différentes pour un même mot, la lemmatisation facilite l'apprentissage des correspondances entre l'anglais et le français. Cela est particulièrement utile dans un contexte où les ressources sont limitées, car le modèle est moins contraint par les variations morphologiques et peut mieux généraliser les relations linguistiques entre les deux langues.

Cependant, cette simplification entraîne également une perte d'information grammaticale. Par exemple, la suppression des flexions verbales ou nominales peut rendre plus difficile la reconstruction correcte des phrases en sortie, ce qui peut impacter négativement la qualité de la traduction dans certains cas. Nous allons maintenant lemmatiser nos corpus anglais et français afin d'observer les conséquences de la lemmatisation sur la traduction.

1. Le lemmatiseur NLTK pour l'anglais

La lemmatisation en anglais a été réalisée à l'aide de la bibliothèque SpaCy avec son modèle de langue en anglais. Ce modèle permet de traiter les mots en fonction de leur contexte grammatical et de réduire les formes fléchies à leur forme de base (lemme). Contrairement au WordNetLemmatizer de NLTK, qui utilise la base lexicale WordNet, le modèle de SpaCy est conçu pour être plus rapide et précis dans le traitement des phrases complètes. Il lemmatise chaque mot en tenant compte de son rôle dans la phrase, ce qui permet d'obtenir des lemmes de manière contextuellement appropriée.

2. Le lemmatiseur NLTK pour le français

Pour la lemmatisation du français, nous avons utilisé le modèle de SpaCy pour la langue française, qui offre un traitement similaire à celui du modèle anglais, mais adapté aux particularités de la langue française. Le modèle fr_core_news_sm de SpaCy est capable de reconnaître les variations morphologiques des mots en français, telles que les conjugaisons verbales ou les accords de genre et de nombre. En analysant le contexte grammatical de chaque mot, il génère son lemme de manière précise, tout en respectant les spécificités grammaticales de la langue.

3. Analyse des scores BLEU et comparaison avec les formes fléchies

Nous avons mis en place un script **lemmatisation.py** permettant de traiter les corpus parallèles en anglais et en français. Ce script utilise WordNetLemmatizer pour l'anglais et SpaCy pour le français. Le processus de lemmatisation consiste à transformer chaque mot fléchi en son lemme tout en conservant la structure du corpus (une phrase par ligne). Tout d'abord, on télécharge les dépendances nécessaires. Voici comment exécuter le script :

pip install nltk
pip install nltk spacy
python -m spacy download en_core_web_sm
python -m spacy download fr_core_news_sm
pip install git+https://github.com/ClaudeCoulombe/FrenchLefffLemmatizer.git

python lemmatization.py

Nous avons généré de nouveaux fichiers de configuration **lemmes1.yaml** et **lemmes2.yaml** pour inclure les corpus lemmatisés. Les phases d'entraînement ont été relancées en suivant le même

protocole que pour l'exercice avec les formes fléchies ci-dessus. Nous obtenons les résultats suivants pour les deux runs effectuées :

Run	Corpus	Domaine	Score BLEU
Run n°1	Europarl	Domaine	18,67
Run n°1	EMEA	Hors domaine	0,36
Run n°2	Europarl + EMEA	Domaine	9,93
Run n°2	EMEA	Hors domaine	77,38

L'évaluation du moteur de traduction neuronale OpenNMT sur un corpus en lemmes met en évidence des changements notables par rapport à l'évaluation réalisée sur les formes fléchies. La lemmatisation modifie la manière dont le modèle apprend et génère les traductions, ce qui se reflète dans les scores BLEU obtenus ci-dessus.

L'une des premières observations concerne la baisse des performances sur le corpus Europarl. Lorsque l'on compare avec les résultats obtenus en formes fléchies, on constate une diminution du score BLEU, qui passe de 25,82 à 18,67 dans la run n°1 et de 14,36 à 9,93 dans la run n°2.
Cette perte s'explique par la disparition des flexions grammaticales, qui jouent un rôle clé dans la structure syntaxique et la cohérence du texte traduit. La lemmatisation entraîne une simplification du vocabulaire qui, si elle réduit la complexité de l'apprentissage, prive aussi le modèle d'informations essentielles pour produire des phrases grammaticalement correctes.

L'absence de distinctions morphologiques entre les différentes formes d'un même mot complique la réinterprétation du contexte, ce qui affecte directement la fluidité et la précision des traductions dans un domaine où la diversité lexicale et syntaxique est importante.

En revanche, sur le corpus EMEA, on observe un phénomène contrasté. Lorsque le modèle est uniquement entraîné sur Europarl, les performances hors domaine restent extrêmement faibles, avec un score BLEU de 0,36. Ce résultat confirme que la lemmatisation ne permet pas, à elle seule, d'améliorer la généralisation du modèle vers un domaine spécialisé si celui-ci n'a pas été préalablement exposé aux données de ce domaine.

Au contraire, lorsque le corpus d'entraînement intègre directement des données du domaine médical (Europarl + EMEA), le score BLEU sur EMEA atteint 77,38, soit une amélioration significative par rapport aux 69,17 obtenus avec les formes fléchies. Cette progression s'explique par la nature même des textes spécialisés : dans un corpus technique, la terminologie est souvent stable et moins sujette aux variations morphologiques.
La lemmatisation favorise alors une meilleure correspondance entre les termes anglais et français en réduisant les formes alternatives et en renforçant la cohérence du vocabulaire. Ainsi, en supprimant les flexions verbales et nominales, le modèle peut se concentrer sur les correspondances directes entre mots-clés, ce qui améliore la qualité des traductions dans un contexte spécifique.

Ces résultats montrent que la lemmatisation a un effet contrasté selon le type de corpus utilisé. Dans un contexte généraliste comme Europarl, la perte d'informations grammaticales nuit à la fluidité et à la précision des traductions. Le modèle, privé des indices morphologiques nécessaires à une reconstruction fidèle de la phrase, produit des résultats plus approximatifs. À l'inverse, dans un domaine spécialisé comme EMEA, où la terminologie est plus rigide et la diversité morphologique

moins importante, la lemmatisation permet au modèle d'être plus efficace en limitant les variations inutiles et en stabilisant le vocabulaire.

IV. Points forts du moteur OpenNMT

L'un des principaux avantages d'OpenNMT réside dans sa flexibilité et sa facilité d'utilisation. En effet, la plateforme prend en charge diverses architectures de modèles neuronaux, telles que les réseaux de neurones récurrents (LSTM) et les Transformers, qui sont parmi les plus efficaces pour la traduction automatique. Cette large étendue pour différents types de modèles permet de choisir celui qui est le mieux adapté à notre situation. En outre, OpenNMT est bien documenté, ce qui en fait un outil très accessible, même pour nous qui n'y connaissions rien avant de commencer ce projet.

De plus, son interface est simple et permet de commencer facilement avec des configurations par défaut tout en offrant la possibilité de personnaliser en profondeur le processus d'entraînement et de traduction. Les outils de prétraitement et de post-traitement des données sont également bien intégrés, ce qui permet de manipuler facilement les corpus de données bilingues. Des outils comme le tokenizer, le truecaser et le clean-corpus sont facilement accessibles, permettant un travail de préparation des données rapide et efficace. Cela facilite la gestion des corpus volumineux comme Europarl et la mise en place des processus nécessaires pour assurer une bonne qualité des traductions.

Finalement, OpenNMT offre une bonne prise en charge du calcul parallèle, ce qui permet d'entraîner des modèles sur GPU, accélérant ainsi le processus d'entraînement. Le moteur est également très performant en termes de vitesse d'entraînement sur des corpus de taille modérée, ce qui est un atout important pour les projets à court terme ou les essais rapides.

V. Limitations d'OpenNMT et difficultés rencontrées

Cependant, malgré ces avantages, OpenNMT présente plusieurs limitations qui méritent d'être abordées. L'une des plus grandes difficultés rencontrées concerne la durée d'entraînement des modèles sur des corpus volumineux comme Europarl.
En effet, bien que le moteur soit optimisé pour l'entraînement sur GPU, il reste relativement lent sur des machines dépourvues de cette capacité (comme les ordinateurs de Polytech), limitant ainsi l'accès au système pour ceux qui ne disposent pas d'un matériel performant.

Par exemple, dans le cadre de l'utilisation des ordinateurs mis à disposition par l'école et pour un entraînement sur un modèle de traduction, le processus a pris plusieurs heures, malgré l'utilisation de GPU. Bien que l'utilisation de Google Colab ait permis de bénéficier d'un accès à des ressources GPU et donc de réduire ce temps, le processus d'entraînement s'est avéré être encore extrêmement long, ce qui a conduit à des attentes prolongées avant d'obtenir des résultats. Cela a non seulement ralenti le flux de travail, mais a également conduit à des tests et expérimentations limités, car le temps d'attente entre chaque cycle d'entraînement était trop long pour effectuer des ajustements rapides.

Une autre limitation constatée lors des expérimentations est liée à la capacité du modèle à traiter de grands corpus ou des corpus spécialisés, comme ceux du domaine médical. Lorsque le modèle est confronté à des termes techniques ou des structures syntaxiques spécifiques, la qualité de la traduction diminue, ce qui se reflète dans des scores BLEU plus faibles. Cette limitation pourrait être liée à un manque de données d'entraînement spécifiques au domaine ou à un modèle trop général qui n'a pas été suffisamment adapté à ces particularités.

En somme, OpenNMT présente des avantages indéniables en termes de flexibilité et de performance, mais des difficultés liées au temps d'entraînement et à la gestion des données peuvent être des obstacles significatifs, en particulier dans des projets d'envergure nécessitant de multiples ajustements et validations, comme dans notre cas, puisque nous ne connaissions pas les bonnes valeurs à

prendre, notamment pour valid_steps, et où il a fallu mesurer différents scores BLEU pour le déterminer.

VI. Pistes pour un travail futur

Pour résoudre certaines de ces limitations, plusieurs pistes peuvent être explorées dans un futur travail. Tout d'abord, il serait pertinent de tester l'ajout de données spécifiques au domaine pour entraîner un modèle plus spécialisé, en particulier dans les domaines techniques comme la médecine. L'augmentation de la diversité et de la quantité des données d'entraînement pourrait permettre au modèle d'améliorer sa capacité à traduire des termes et des structures spécifiques.

Une autre solution consisterait à tester l'implémentation de modèles plus complexes, comme les transformers multilingues, qui peuvent mieux gérer les différences entre les langues tout en offrant des performances accrues sur des corpus de grande taille. Il serait également intéressant de travailler sur l'amélioration du prétraitement, notamment avec une meilleure gestion des lemmes et des formes fléchées, pour améliorer la qualité de la traduction sur des corpus flexionnels.

Enfin, pour résoudre les problèmes de performance sur des machines sans GPU, il serait utile de se pencher sur l'optimisation de l'entraînement et de la traduction, par exemple en utilisant des techniques de distillation de modèle pour réduire la taille des modèles tout en préservant leur efficacité.

VII. Organisation

Puisque nous débutions toutes les deux dans ce domaine, nous avons pris la décision de tout faire ensemble, de l'installation des outils à la rédaction de ce rapport, afin de mieux comprendre les étapes et les concepts liés à la traduction automatique avec OpenNMT.

Comme le travail demandait beaucoup de temps, nous avons juste choisi de nous diviser les mesures. Léa s'est occupée des mesures des formes lemmatisées, tandis que Marie s'est occupée des formes fléchies, ce qui nous a permis de réaliser chaque mesure plus rapidement, étant donné que chaque mesure prenait environ 4 heures.

Concernant le projet sur GitHub, nous avons utilisé cette plateforme pour centraliser notre code, nos modèles et nos résultats. Nous avons veillé à bien organiser les fichiers pour garantir la clarté du travail. Le code du projet était partagé entre nous, avec une gestion commune des différentes étapes d'entraînement et des tests de modèles. Tous les codes sont disponibles sur le lien du Github :

https://github.com/LeaMagloireLaGreve/TAL_Marie_CRANSAC_Lea_MAGLOIRE_LA_GREVE