Institut National des Langues et Civilisations Orientales

Département Textes, Informatique, Multilinguisme

Intégration des technologies de traduction automatique neuronale à l'échelle d'une agence de traduction

Master

Traitement automatique des langues

Parcours :

Ingénierie Multilingue

Par

Victorien VILLIERS

Encadrant :

Pascal Trouvin

Année universitaire 2018-2019

Table des matières

Table des illustrations

Table des tableaux

Remerciements

J'adresse mes remerciements à l'ensemble de l'équipe pédagogique du master Plurital, qui m'a permis d'acquérir les compétences et les connaissances nécessaires à l'écriture de ce mémoire, en particulier l'équipe ERTIM et les enseignants-chercheurs du parcours Ingénie Multilingue.

Je tenais également à remercier l'équipe d'Univoice qui m'a permis de mener à bien les différentes études décrites dans le présent document, d'acquérir une expérience pratique de la traduction automatique et de participer au processus de développement.

Résumé

Le présent mémoire compile deux études menées au sein de l'agence de traduction Univoice, en lien avec la traduction automatique neuronale. Dans la première, nous évaluons les traductions produites par différents outils de traduction automatique neuronale disponibles sur le marché pour la paire de langue anglais-français, dans le cadre de la sélection d'un fournisseur tiers. Nous nous aidons pour cela de deux métriques humaines, inspirées du standard MQM-DQF, et de la métrique automatique BLEU. Dans la seconde étude, nous évaluons la viabilité du développement de moteurs de traduction automatique neuronale en interne, à l'aide de la technologie OpenNMT et de corpus multilingues alignés open-source. Nous mettons en place une série de scripts permettant l'extraction de textes alignés à partir de mémoires de traduction au format TMX et l'entraînement de modèles basés sur la technologie des réseaux de neurones récurrents LSTM, et évaluons les résultats obtenus sur un échantillon de traductions générées en sortie.

Mots-clés : marché de la traduction, traduction automatique neuronale, métriques d'évaluation, BLEU, MQM-DQF, OpenNMT, corpus multilingues alignés, TMX, réseau de neurones récurrents LSTM

Abstract

This paper gathers two studies lead within the translation agency Univoice in the field of neural machine translation. In the first study, we analyze the translations generated by several neural machine translation tools available in the market for the English-French language pair, in the context of selecting a third-provider. For the purpose of this analysis, we use two human metrics inspired from the MQM-DQF standard, and the automatic metric BLEU. In the second study, we analyze the viability of developing in-house neural machine translation engines using OpenNMT toolkit and open source aligned multilingual corpora. For this purpose, we define several scripts aimed at the extraction of aligned texts from TMX translation memories, and the training of translation models based on long short-term memory recurrent neural networks. We then evaluate a sample of the translations produced by the model.

Keywords: translation market, neural machine translation, evaluation metrics, BLEU, MQM-DQF, OpenNMT, aligned multilingual corpora, TMX, LSTM recurrent neural networks

Introduction

Avant l'apparition des technologies numériques, la traduction est restée pendant longtemps une discipline essentiellement artisanale. L'internationalisation croissante des publications, l'accélération des échanges et la diffusion de contenus à très grande échelle, n'ont eu cesse d'augmenter la demande en traductions, que ce soit pour l'adaptation de textes juridiques, d'œuvres culturelles ou la commercialisation de produits à l'international.

La démocratisation de l'Internet grand public dans les années 90 et l'apparition des premiers outils d'aide à la traduction ont accéléré la transformation du secteur, passant d'une traduction artisanale à une production « industrielle ». La volonté de produire rapidement des contenus dans toutes les régions du globe, tout en touchant un maximum de publics dans une langue qui leur ait familière, a fait du traducteur un intervenant clé de la mondialisation et de la production à grande échelle de produits manufacturés, de logiciels et de sites Web.

Aujourd'hui, la traduction automatique et les technologies d'apprentissage, en particulier le deep learning et la traduction automatique « neuronale », changent une nouvelle fois la donne en industrialisant davantage le métier, diminuant peu à peu l'effort de traduction, et reléguant le traducteur au rang de relecteur de traduction machine, garant de l'intelligibilité des traductions produites par des algorithmes.

C'est dans ce contexte d'industrialisation de la traduction et de démocratisation des technologies de traduction automatique que s'inscrivent les travaux présentés dans ce mémoire. Dans l'écosystème actuel de la traduction et au vu de l'évolution du marché, nous nous demanderons comment une agence de traduction peut, à son échelle, intégrer les technologies de traduction automatique neuronale disponibles à son portefeuille technologique, afin de garder une place concurrentielle et de s'affranchir des outils souvent imposés par ses donneurs d'ordre, et ce, avec des moyens techniques et budgétaires restreints.

Afin de répondre à cette problématique, nous nous appuierons sur deux études de cas menées au sein de l'agence de traduction Univoice. Dans la première, nous nous intéresserons à l'évaluation des outils de traduction automatique neuronale disponibles sur le marché pour la sélection d'un fournisseur tiers, en nous interrogeant notamment sur la pertinence des diverses métriques d'évaluation. Dans la seconde, nous étudierons la possibilité d'intégrer cette technologie via le développement de moteurs de traduction automatique neuronale en interne grâce à la technologie open source OpenNMT, et évaluerons les résultats obtenus sur un premier échantillon spécialisé.

1. Présentation des concepts clés

Afin de rendre le reste de cet exposé plus accessible, nous avons jugé utile de revenir sur quelques notions essentielles.

TM (Translation Memory, mémoire de traduction) : Document structuré (généralement au format XML), permettant de stocker les traductions sous forme d'unités de traduction (TU : Translation Unit), elles-mêmes formées de plusieurs segments multilingues alignés. Le standard le plus utilisé est le format TMX (Open Standards for Container/Content Allowing Re-use (LISA), 1997). Les mémoires de traduction sont généralement typées par langues et par domaine. Elles servent de données d'entrée pour l'entraînement des moteurs de traduction automatique.

XLIFF (XML Localization Interchange File Format) : Format standard s'appuyant sur le standard TMX et utilisé pour stocker des documents de traduction. Comme dans le format TMX, les phrases sont stockées sous formes d'unités de traduction (TU : Translation Unit), elles-mêmes composées de segments dans plusieurs langues. (Organization for the Advancement of Structured Information Standards XLIFF TC, 2008)

Traduction humaine : Traduction réalisée par un traducteur humain, sans traduction automatique préalable. Le terme « traduction humaine » (HT, « Human Translation ») est entré dans l'usage par opposition à la « traduction automatique » ou « machine » (MT, « Machine translation »).

TAO (Traduction assistée par ordinateur) : Traduction humaine réalisée à l'aide d'outils de gestion de mémoires de traduction (outils TAO). Ces outils permettent de segmenter le texte source en phrases (ou « segments ») et de conserver les traductions effectuées par des traducteurs humains. Le traducteur peut récupérer automatiquement la traduction d'une phrase déjà stockée en mémoire, pour un gain de temps et de cohérence. De même, si la traduction n'est pas exactement identique, l'outil déterminera la distance d'édition avec la phrase existante, afin de lui attribuer un score de similarité (de 0 à 100%), et mettre en évidence les différences entre les deux phrases. Les outils de traduction assistée par ordinateur réunissent souvent toute une suite de fonctionnalités annexes, telles que la correction orthographique, ou l'ajout de glossaires et de dictionnaires.

Traduction automatique (ou MT, « Machine Translation ») : Traduction réalisée à l'aide d'un algorithme. Le contenu prétraduit peut être soumis à un traducteur humain, qui s'emploiera à valider, améliorer ou corriger le contenu prétraduit. Ce processus de validation est appelé « post-édition » (PE). Les performances du moteur peuvent être évaluées en fonction du temps passé par le traducteur pour valider ou corriger la traduction, ou le nombre de corrections apportées. Il existe trois grandes familles de moteurs de traduction automatique : à base de règles, statistique et neuronale. (Systran Blog, 2016)

NMT (Neural machine translation, traduction automatique neuronale) : La traduction automatique « neuronale », que nous étudierons plus en avant dans le cadre de ce mémoire, s'appuie sur la technologie des réseaux de neurones. Elle est à ce jour la technologie de traduction automatique la plus performante et la plus utilisée sur le marché de la traduction, supplantant largement la traduction automatique à base de règle et la traduction automatique statistique pour les langues de grande diffusion.

2. Présentation de l'environnement de travail

Mon stage s'est déroulé sur deux sessions de deux mois, une première à l'issue de ma première année de master, puis une seconde à l'issue de ma deuxième année. J'ai eu l'opportunité d'effectuer ce stage dans l'agence de traduction qui m'embauchait jusqu'alors en tant que traducteur, responsable qualité et chef de projet. Ce mémoire s'inscrit dans une évolution pluridisciplinaire, à la croisée des métiers de la traduction, du TAL et du machine learning. Dans cette première partie, nous commencerons par décrire l'entreprise et sa place sur le marché de la traduction, avant d'expliquer dans le détail le projet dans lequel s'inscriront nos deux études mentionnées en introduction.

2.1 L'entreprise

Univoice est une agence de traduction traduisant principalement des projets de l'anglais vers le français, et dans une moindre mesure depuis et vers d'autres langues de grande diffusion. Ouverte en 2007, la PME se spécialise depuis ses débuts dans le domaine de la localisation (adaptation au marché français) de logiciels et de sites web, et dans la traduction de documents techniques.

Depuis sa transformation en SAS en 2017, ses activités se sont diversifiées avec l'arrivée de nouveaux projets en post-édition, permettant d'accroître le volume de mots traités. En 2018, Univoice traitait plus de 11 millions de mots en traduction, dont 4,6 millions en post-édition. La plupart des projets traités provenaient de grands comptes du secteur informatique (logiciel et matériel) et des majors du Web.

La collaboration avec son entreprise sœur, O4S, spécialisée le développement et la sécurité informatique, a permis l'ouverture d'un pôle développement en 2019 pour la conception et le déploiement d'un marketplace, permettant d'appairer automatiquement les compétences des traducteurs avec les demandes de clients directs. Grâce à ce nouvel outil, l'entreprise souhaite diversifier son activité en couvrant davantage de combinaisons langues et de domaines de spécialité, et augmenter à terme sa base de clients directs et de partenaires, en France comme à l'étranger.

Depuis 2018, l'entreprise souhaite également prendre le tournant de la traduction automatique neuronale, point que nous traiterons plus en avant dans le cadre de ce mémoire. L'objectif à terme de toutes ces évolutions est de diminuer la dépendance de l'entreprise aux donneurs d'ordre et aux outils tiers, de garder le contrôle sur ses données face la « cloudification » des données et des outils, et de proposer de son propre outil de gestion des projets, contrôlable et centralisé.

2.1.1 L'équipe

En mai 2019, l'équipe se composait de quinze employés répartis en trois pôles : direction, traduction et informatique.

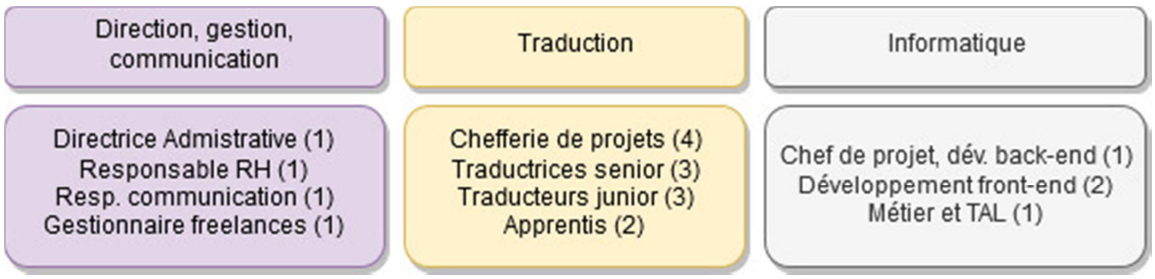


Figure 1 : *Répartition des différents métiers au sein d'Univoice*

2.1.2 Place de l'entreprise sur le marché de la traduction

On décompose traditionnellement le marché de la traduction en trois catégories d'acteurs (Risku, 2007) :

1. Les agences de traduction dites « multilingues » (ou « MLV » en anglais, pour « Multilingual Vendor ») comme SDL ou Lionbridge, capables de gérer les projets de traduction dans plusieurs combinaisons de langues grâce à un réseau de plusieurs agences monolingues et de traducteurs indépendants. Ces dernières s'occupent généralement du prétraitement des fichiers à l'aide d'outils TAO et de moteurs de traduction automatique, et imposent souvent à leurs fournisseurs leur technologie.

2. Les agences de traduction dites « monolingues » (ou « SLV » en anglais, pour « Single-Language Vendor ») comme Univoice ou encore Rheinschrift, proposant des services de traduction dans une seule ou quelques combinaisons de langues, et collaborant avec un pool de traducteurs (ces dernières sont généralement spécialisées dans un ou plusieurs domaines tel que l'informatique, le médical ou encore le juridique).

3. Les traducteurs indépendants (ou « freelances »), en bout de chaîne, traduisant généralement d'une à deux langues vers leur langue maternelle.

Malgré cette organisation pyramidale, il n'est pas rare qu'un traducteur indépendant ou une agence de traduction monolingue gère également des demandes de clients directs. Ces derniers sont d'ailleurs souvent recherchés, car plus rentables. La gestion de demandes de clients directs supposent néanmoins un éventail de compétences plus large (mise en page, la manipulation des fichiers, la gestion des documents de référence et terminologique, la gestion d'outils TAO et des mémoires, pour ne citer que les principales), en particulier pour les clients grands comptes.

De par ses activités, Univoice se situe dans la catégorie des agences de traduction monolingues, et sert donc principalement d'intermédiaire entre les agences multilingues et les traducteurs indépendants. Même si cette part reste encore faible, elle traite occasionnellement des demandes de clients directs, qu'elle sous-traite la plupart du temps à d'autres agences dans le cas de combinaisons de langues ou de domaines pour lesquels elle ne dispose pas des compétences suffisantes.

On peut représenter la configuration actuelle et la place d'Univoice sur le marché de la manière suivante :

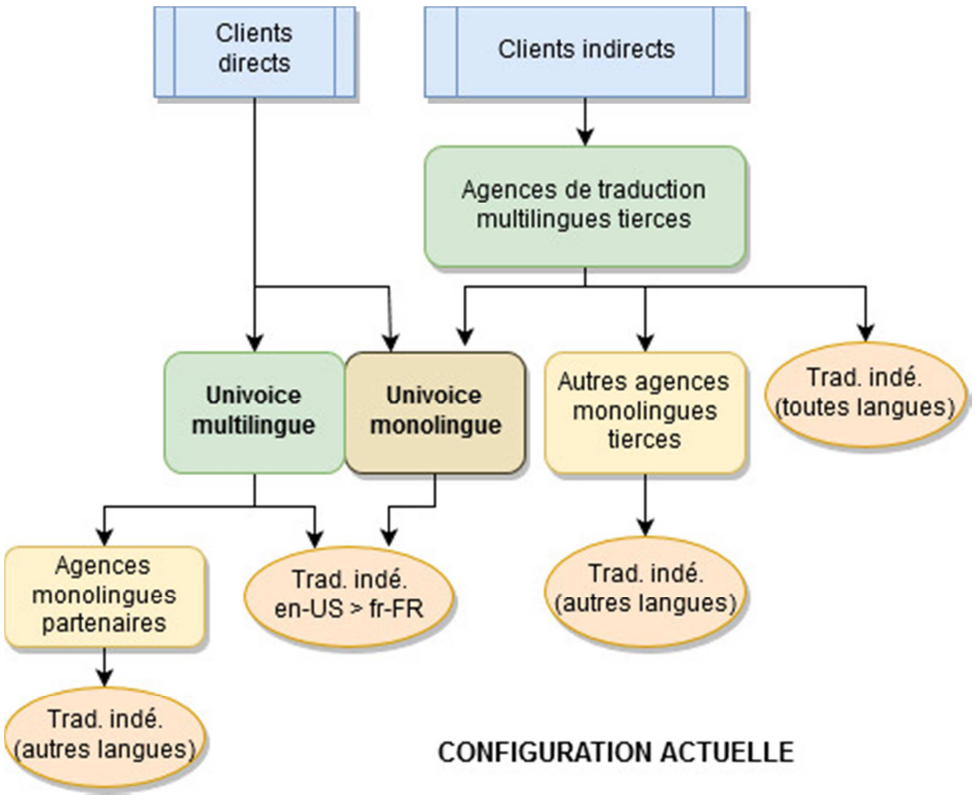


Figure 2 : *Configuration actuelle : place d'Univoice sur le marché de la traduction*

Cette configuration est aujourd'hui court-circuitée par l'arrivée de nouveaux modèles en plein essor, visant à mettre en lien direct entre les clients et les traducteurs, réduisant par la même occasion le temps de traitement des demandes et le coût global. C'est notamment le cas des systèmes de marketplace spécialisés, tels que ceux de Systran, Gengo, Lingotek, ou génériques, tels qu'Amazon Mechanical Turk, permettant des traductions à la demande, ou encore des plateformes de traduction basées sur le crowdsourcing, développées principalement par les géants du Web en réponse à leurs besoins en traductions dans de nombreuses langues (application Translate Facebook, Twitter Translation Center).

2.2 Le projet

Notre étude s'inscrit dans le cadre du développement d'un système de marketplace spécialisé, initiée en 2018 par la société Univoice. Dans cette partie, nous verrons le fonctionnement général de la plateforme (objectifs, architecture, modélisation des processus et rapprochement projet-compétence), ainsi que la manière dont sont extraits les segments destinés à la traduction automatique.

Projet	Plateforme marketplace
Objectifs	*-> Proposer une plateforme mettant en relation les demandes de traduction (clients) et les compétences des traducteurs*

Équipe	- 1 développeur back-end - 2 développeurs front-end - 1 taliste, référent métier
Technologie	- Front : Typescript, Angular - Back : Python, Flask - TAL : Python, NLTK
Tâches	- Modéliser le workflow de gestion de projet et de validation des compétences des traducteurs - Modéliser les documents en entrée - Prétraiter les données en entrée pour traduction

2.2.1 Objectif et fonctionnement général

Le principal objectif de la plateforme est de mettre en relation les demandes de traduction de clients et les compétences des traducteurs. Côté client, elle permet de déposer des fichiers à traduire directement depuis un navigateur, et de les soumettre à un pool de traducteurs compétents. Côté traducteurs, elle permet de recevoir des traductions correspondant à leur domaine de spécialité et à leur combinaison de langues. Côté chef de projets, elle permet de suivre les projets et d'interagir avec les clients et les traducteurs grâce à un système d'alertes.

La plateforme propose donc un fonctionnement double : d'une part celle d'un marketplace classique visant à créer un canal direct entre les demandes des clients et les prestataires de traduction et, d'autre part, celle d'un système TMS (Translation Maganement System[1]) en centralisant le processus de gestion de projets et de manipulation des documents.

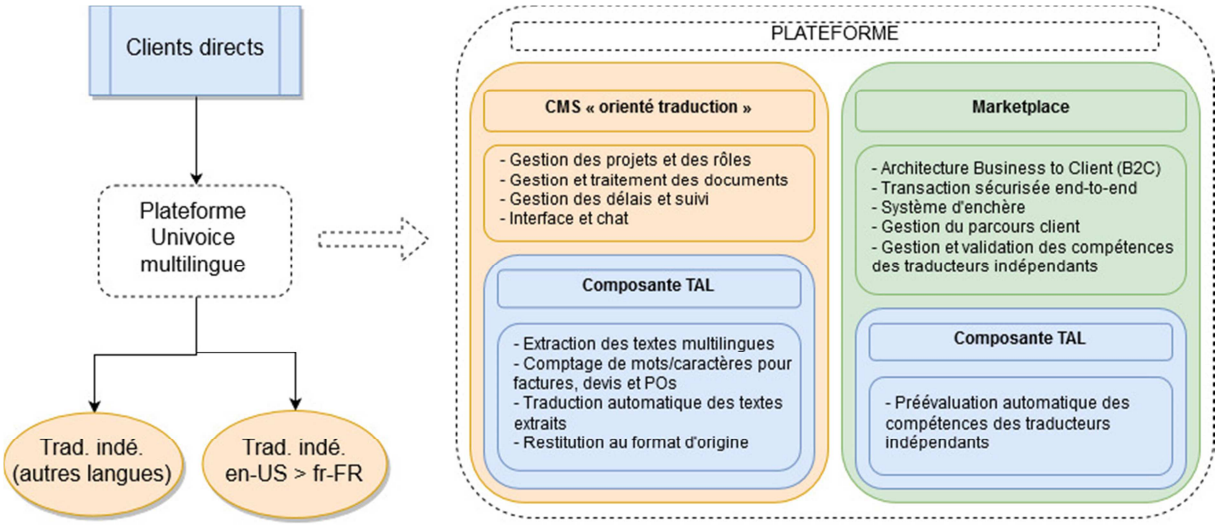


Figure 3 : *Nouvelle configuration et composantes de la plateforme*

[1] https://en.wikipedia.org/wiki/Translation_management_system

2.2.2 Architecture globale de la plateforme

La plateforme s'appuie sur une architecture en containers basée sur la technologie Docker[2]. Ce type d'architecture sous forme de microservices est utile pour isoler les différents programmes d'une application, et accroître sa flexibilité et sa portabilité. Son aspect modulaire permet également aux équipes de travailler indépendamment sur chaque container tout en poursuivant un développement agile et continu, contrairement aux architectures monolithiques qui nécessitent un redéploiement complet de la solution.

La plateforme présente une architecture classique front-end/back-end adossée à une base de données. Les différents containers, tels que les containers dédiés aux solutions de traduction automatique, ou à d'autres traitements TAL, sont appelés par le container back-end principal codé en Python et basé sur le framework Flask[3], à l'aide d'un système de file d'attente de type Redis queue[4].

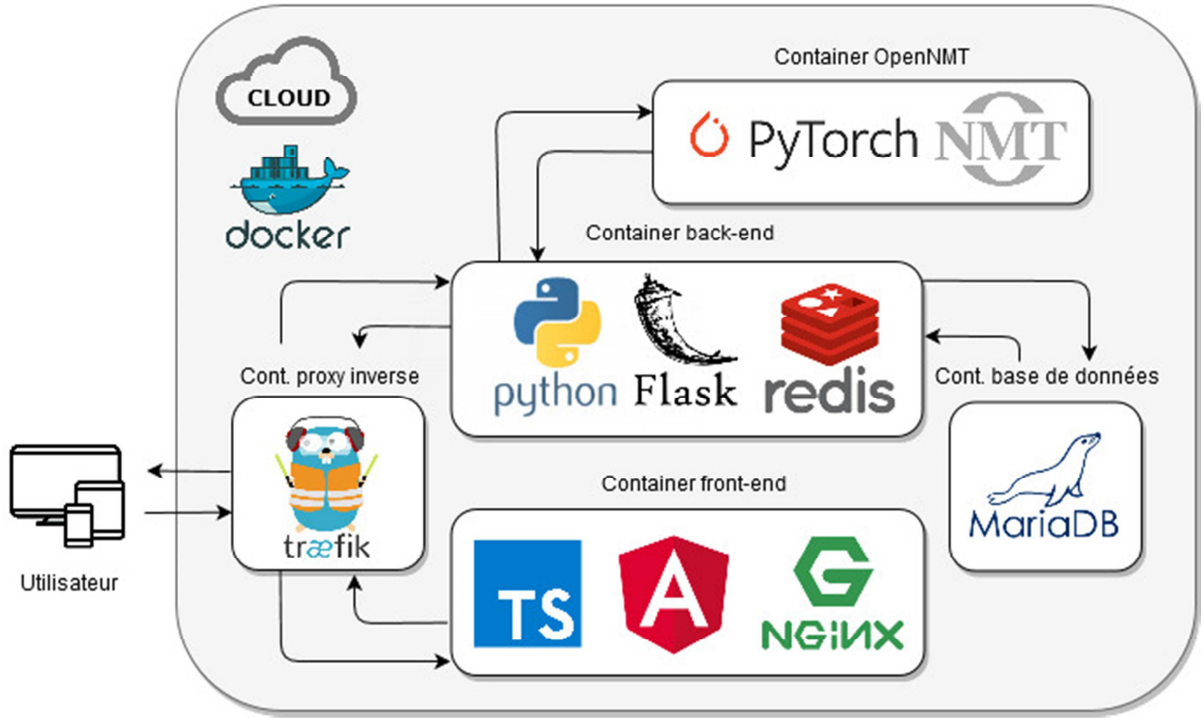


Figure 4 : *Schéma de l'architecture en containers de la plateforme*

2.2.3 Modélisation du workflow de projets et de validation des compétences

Dans la pratique, le parcours d'un projet de traduction suit toujours le même schéma : un chef de projet reçoit la demande d'un client, récupère les informations essentielles au bon déroulement du projet, analyse la faisabilité en fonction des traducteurs disponibles, et envoie la traduction en production. Le traducteur qui reçoit le projet doit alors produire sa traduction dans les

[2] https://docs.docker.com/
[3] https://flask.palletsprojects.com/en/1.1.x/
[4] https://redis.io/documentation

contraintes imposées (style, documents de référence, délais). Le document traduit passe par une seconde phase d'analyse qualité avant livraison, afin de garantir une qualité finale optimale.

La plateforme modélise ce workflow sous forme de rôles et de statuts. Le workflow complet d'un projet est décrit dans l'annexe I du présent mémoire. Côté traducteurs indépendants, la plateforme modélise également le processus de validation des compétences, préliminaire à l'ajout d'un traducteur indépendant à la base de données des partenaires. Le workflow de validation de compétences est décrit dans l'annexe II du présent mémoire.

La modélisation suppose une connaissance et une compréhension fine des caractéristiques des compétences en traduction. Elle nécessite notamment de s'appuyer sur l'expertise des différents acteurs métiers, afin de s'assurer que les données sont pertinentes et représentatives. Cette étape a en effet un impact direct sur l'organisation de la base de données et sur l'utilisabilité de l'application. Par exemple, il convient de bien distinguer langue source et langue cible, le sens des traductions n'étant pas bidirectionnel, ou encore de bien étudier la catégorisation et la granularité des domaines, afin de ne pas augmenter inutilement le nombre de compétences disponibles.

2.2.4 Rapprochement des projets et des compétences

Les projets et les compétences des traducteurs sont stockés en mémoire sous forme de jeux de données. Ces derniers sont utilisés afin de mettre en lien les demandes des clients avec les compétences des traducteurs. Ils servent également en aval à regrouper des contenus traduits similaires au sein de mémoires de traduction homogènes qui serviront, a posteriori, à entraîner et à paramétrer les moteurs de traduction automatique.

Chaque projet se compose d'un ou plusieurs fichiers à traduire, accompagné d'un ensemble de métadonnées. Ils sont appariés selon le domaine de spécialité, la langue source et la langue cible du traducteur.

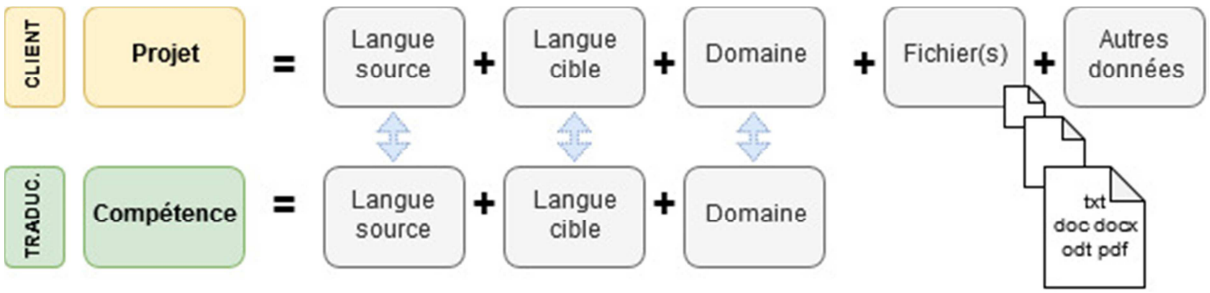


Figure 5 : *Schéma de représentation des données : Rapprochement projets-compétences et types de fichiers en entrée*

2.2.5 Extraction des chaînes pour traduction automatique

Avant d'être envoyé en traduction automatique, les fichiers soumis par le client passent par une série de prétraitements. La première étape consiste à extraire le texte du fichier. L'extraction est effectuée au moyen la bibliothèque Python tierce proposant des fonctionnalités d'extraction dans les formats texte les plus répandus : doc, docx, pdf, odt. Cette opération va déstructurer le fichier

original afin d'en extraire le texte et la structure. La structure est conservée afin de procéder à la mise en forme du document lors de la restitution finale du texte dans son format d'origine.

Une fois le texte extrait, ce dernier passe par une étape de segmentation (partition en segments/phrases) à la manière d'un outil TAO, afin qu'il puisse être utilisé par le traducteur. Cette étape est indispensable, car ce seront les segments extraits qui sont soumis un à un au moteur de traduction automatique. Les textes sont stockés au format XLIFF. Un compteur de mots est également implémenté afin de calculer le devis des projets.

Une fois le texte segmenté, chaque segment du fichier XLIFF est envoyé à un moteur de traduction automatique. On peut représenter le processus de la manière suivante.

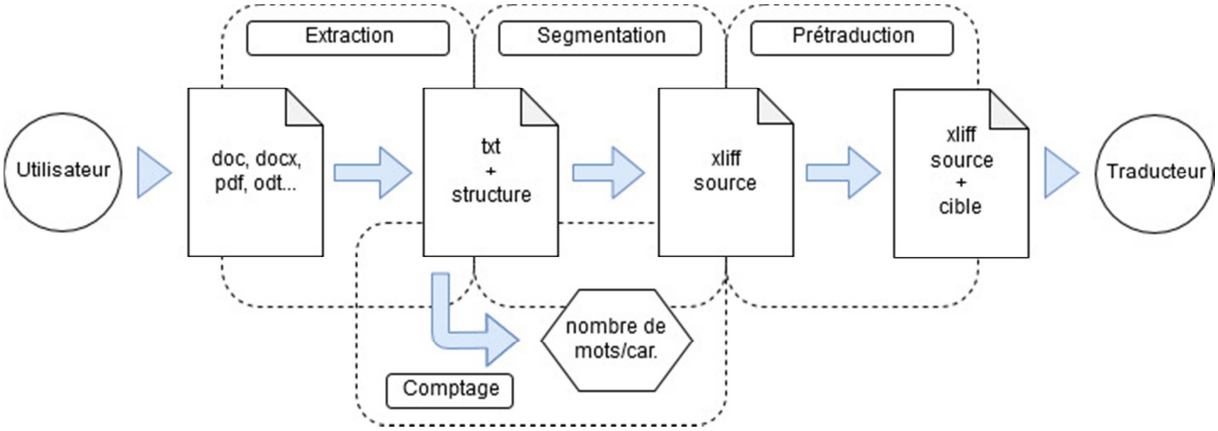


Figure 6 : *Schéma du processus de prétraitement des fichiers*

Pour la prétraduction des textes, deux stratégies ont été adoptées : une première, plus simple à mettre en œuvre et implémentée dans la version actuelle, qui s'appuie sur un fournisseur de traduction automatique tiers via une API, et une seconde, actuellement en développement, qui s'appuie sur des moteurs de traduction automatique entraînés en interne à l'aide de la technologie OpenNMT. Ces composantes seront abordées plus en détail dans nos deux études.

2.2.6 Spécificités liées au prétraitement

L'extraction du texte suppose de connaître l'extension du fichier en entrée, cette donnée est vérifiée en analysant la cohérence entre l'extension du nom du fichier, son type MIME et le type indiqué dans la structure du fichier. Cette étape permet de rediriger le fichier vers la fonction adéquate au sein de la bibliothèque d'extraction. Les fichiers PDF présentent par exemple une spécificité : les PDF de type « image » nécessitent de recourir à la reconnaissance optique des caractères, tandis que les PDF intégrant directement des contenus textuels dans leur structure peuvent être extraits sans étape intermédiaire.

La segmentation en phrases et le passage au format XLIFF sont réalisés à l'aide d'un programme maison partitionnant les textes grâce aux caractères de retour à la ligne présents dans le texte et aux points finaux. La limite de cette méthode est bien entendu les caractères de retour à la ligne de « mise en page », en particulier ceux pouvant apparaître suite à l'extraction de textes au

format PDF ainsi que les points ne marquant pas la fin d'une phrase. Nous avons donc dû implémenter des règles supplémentaires afin de prendre en compte tous les cas de figure.

Le comptage en mots est réalisé grâce à la fonction de tokenisation de la bibliothèque NLTK[5]. Le comptage est réalisé sur les espaces, comme dans le comptage dans Word, sans expression régulière particulière pour la reconnaissance de motifs. Le comptage doit également tenir compte de la langue analysée, car s'il est pertinent de compter le nombre de mots pour la facturation des langues alphabétiques, les langues utilisant des syllabaires et/ou des caractères, comme le japonais, le coréen ou le chinois, nécessitent un autre mode de comptage basé sur le nombre de caractères.

Il convient de souligner que toutes ces étapes dépendent de la langue du document en entrée et en sortie, et ne sauraient être appliquées à l'ensemble des langues traitées par la plateforme. Chaque module devra donc être développé et perfectionné pour chaque langue.

2.2.7 Conclusion : Environnement de travail

Comme nous l'avons vu dans cette première partie introductive, Univoice est une agence de traduction souhaitant développer ses propres outils en interne et tirer parti de la technologie de traduction automatique neuronale. Avec son projet de marketplace, l'entreprise souhaite proposer une solution permettant de mettre en relation les demandeurs et les prestataires de traduction. Cette plateforme offre également des fonctionnalités TMS (Translation Management System) permettant le prétraitement de textes et l'extraction de chaînes à des fins de prétraduction automatique.

Bien que la plateforme se veuille multilingue, la première version se concentre pour l'heure sur la combinaison de langues anglais > français, compte tenu de la dépendance à la langue des outils utilisés. Comme nous l'avons vu, la modélisation des données et des workflows suppose une bonne connaissance du domaine pour assurer un fonctionnement optimal et un rapprochement efficace entre les demandes et les compétences des traducteurs, ainsi qu'une connaissance linguistique des langues traitées.

Dans la partie suivante, nous allons passer en revue les deux études menées pour l'intégration de la technologie NMT au sein de la plateforme. La première concernera l'évaluation des outils NMT disponibles sur le marché à des fins de partenariat éventuel. La seconde s'attardera sur le développement de moteurs de traduction automatique neuronale spécialisés et la viabilité de l'entraînement de tels moteurs en interne.

---

3. Présentation des études

Nous décrirons dans les prochaines parties les deux études menées en lien avec l'acquisition des technologies de traduction automatique neuronale. La première décrira le processus d'évaluation des outils NMT disponibles sur le marché pour le choix d'un fournisseur de traduction automatique neuronale tiers. La seconde décrira le processus d'entraînement de moteurs de traduction automatique neuronale effectué en interne à des fins d'étude de faisabilité.

[5] https://www.nltk.org/api/nltk.tokenize.html

3.1 Étude 1 : étude comparative des systèmes de traduction automatique neuronale pour le choix d'un fournisseur tiers

Nous décrirons dans cette première étude le processus d'évaluation des outils NMT disponibles sur le marché pour le choix d'un fournisseur de traduction automatique neuronale tiers : objectifs, méthodologie employée, outils et échantillons testés, métriques utilisées et résultats d'évaluation.

Étude	**Évaluation des outils de traduction automatique neuronale tiers**
Objectifs	*-> Évaluer les différents fournisseurs de traduction automatique neuronale disponibles sur le marché, potentiellement intégrables à la chaîne de traitement pour la paire de langues EN>FR, à l'aide de métriques d'évaluation*
Équipe	- 1 chef de projet (mise en place des tests, création des échantillons) - 2 traducteurs/relecteurs senior (évaluation humaine)
Technologie	- TAL : Python, NLTK (score BLEU)
Tâches	- Lister les outils disponibles - Créer des tests sur mesure avec échantillons et critères d'évaluation - Évaluer les résultats obtenus

3.1.1 Objectifs et étapes préliminaires

L'objectif de cette première étude est d'analyser et de comparer les différents outils NMT disponibles sur le marché sur plusieurs contenus actuellement traités avec des méthodes de TAO traditionnelles et potentiellement candidats à la traduction automatique, afin de déterminer l'outil le plus performant sur divers types de contenu et d'intégrer à terme l'outil sélectionné à la plateforme.

Cette étude comparative s'est déroulée en amont du projet de développement de la plateforme sur deux mois. Elle fait suite à la production d'un cahier des charges, d'un tableau récapitulatif des outils testés et d'une méthodologie de tests.

3.1.2 Méthodologie

Pour cette première étude, nous nous sommes en partie inspirés de l'étude menée par l'agence Lilt en 2017[6] sur les performances de différents outils de traduction automatique. Pour le choix des outils, nous nous sommes appuyés sur le document de synthèse « Slator 2019 Neural Machine Translation Report : Deploying NMT in Operations »[7]. Nous avons également sélectionné divers échantillons extraits de projets déjà réalisés, afin d'évaluer ces différents outils d'après des

[6] https://labs.lilt.com/2017-machine-translation-quality-evaluation-addendum

[7] https://slator.com/data-research/slator-2019-neural-machine-translation-report-deploying-nmt-in-operations/

critères d'évaluation (deux évaluations humaines et une évaluation automatique avec le score BLEU). Enfin, nous avons tenté une comparaison entre les différents outils à l'aune des résultats obtenus.

3.1.3 Caractéristiques des outils NMT testés

Nous avons retenu sept outils, car ils présentaient les caractéristiques suivantes :

- Moteur de traduction automatique neuronale
- Dispose d'une version dédiée dans la paire anglais > français
- Dispose d'une version d'essai en ligne accessible, gratuite, sans nécessité d'abonnement
- Dispose d'une API permettant de récupérer les traductions, en vue d'une intégration

Tous les outils testés sont actuellement des moteurs NMT dits « génériques » dans la mesure où ils ne permettent pas d'adaptation au domaine du texte. Bien que cela soit aujourd'hui une caractéristique des outils grand public, des recherches sur l'adaptation au domaine offrent aujourd'hui de nouvelles perspectives d'amélioration. (Christophe Servan, 2016). Pour notre étude, nous nous sommes limités aux versions gratuites des moteurs disponibles sur le web par commodité, mais l'étude ne saurait être exhaustive sans tester les résultats sur les traductions produites via l'appel des différentes API.

3.1.4 Échantillons testés

Pour tester les outils retenus, nous nous sommes servis d'échantillons de texte issus de travaux passés, caractéristiques des différents domaines et types de texte les plus traités par Univoice, et ne subissant pour l'heure aucune phase de prétraduction automatique : marketing (présentation et sous-titres), chaînes logicielles et juridique (licence et contrat).

Les exemples ne contiennent ainsi aucune traduction de documents techniques, généralement déjà prétraités en traduction automatique, et offrant généralement des résultats satisfaisants en post-édition. Nous caractérisons les textes selon deux critères : d'une part le « type » de texte de l'échantillon, indiquant la portée générale du texte, avec entre parenthèses la forme spécifique du contenu et, d'autre part, son « domaine », ou thématique générale du texte traduit.

#	Nature	Domaine	Nb seg.	Nb mots	Exemple de segment
1	Juridique (licence)	CAO	20	279	Please read and understand the contents of this agreement before installing or using the software.
2	Marketing (présentation)	CAO	17	338	XXX enables companies to accelerate product innovation and build better products faster by reusing the best designs and replacing assumptions with facts.
3	Marketing (sous-titre)	Sécurité IT	17	529	When we say we enable micro-segmented networks for east-west security, we mean we help banks safeguard their customers' hard earned money.
4	Chaînes logicielles	Scolarité	25	38	Blockquote
5	Juridique (contrat)	CAO	25	247	The PO number should be indicated and referenced in all the correspondence with our suppliers, including their invoices.

Tableau 1 : *Tableau récapitulatif des échantillons testés*

3.1.5 Métriques d'évaluation

Les métriques d'évaluation tentent de mesurer la qualité des traductions produites en sortie. Ces métriques sont essentielles, dans la mesure où elles permettent de déterminer, de manière indirecte, le temps et l'effort nécessaires au post-éditeur pour obtenir un résultat final satisfaisant, et par conséquent la rentabilité des prétraductions NMT. On distingue généralement les métriques d'évaluation en deux catégories : l'évaluation humaine, plus chronophage mais plus représentative de l'effort de post-édition, et l'évaluation automatique, basée sur des calculs de distance, plus simples à implémenter, mais moins représentatives sur certaines catégories de texte. (Bonnie Dorr, 2010)

L'évaluation humaine, effectuée par un traducteur et un expert du domaine, vise à évaluer l'intelligibilité, la fidélité, la fluidité, la pertinence, la compréhension, et la valeur informative. (Aaron Han, 2016) Ces critères étant par nature subjectifs, plusieurs standards ont été mis en place, notamment le modèle MQM-DQF. (German Research Center for Artificial Intelligence (DFKI) and QTLaunchPad, 2015). Les évaluations humaines, malgré leur lourdeur, offrent néanmoins des résultats plus proches de la réalité, dans la mesure où elles permettent de mieux estimer l'effort à fournir en post-édition.

L'évaluation automatique, moins chronophage et moins lourde à implémenter, repose quant à elle sur des calculs de distance entre une phrase traduite par un moteur de traduction automatique et une phrase de référence. Cette nécessité de posséder un corpus de référence la distingue de l'évaluation humaine basée sur des critères de qualité. Ces deux caractéristiques font des métriques automatiques un outil adapté pour l'entraînement des moteurs de traduction automatique, en fournissant un retour stable au mécanisme d'apprentissage.

La métrique la plus utilisée dans la recherche et pour l'entraînement de moteurs de traduction automatique est aujourd'hui la métrique BLEU. Cette métrique sert de standard pour l'entraînement, mais n'est pas la seule utilisée. D'autres métriques basées également sur la distance d'édition telles que la métrique WER, METEOR, GTM, TER, et HTER sont également employées dans une moindre mesure. (Bonnie Dorr, 2010) On constate que les métriques les plus performantes sont celles basées sur une moyenne pondérée de plusieurs métriques. (Voir Annexe V : Récapitulatif des métriques d'évaluation pour la traduction automatique.)

Pour notre étude, nous nous sommes basés sur une approche pragmatique et des métriques personnalisées, afin d'évaluer rapidement les traductions produites à l'aide de nos ressources en interne, et de proposer une analyse proche de la réalité. Nous nous sommes appuyés pour cela sur deux évaluations humaines (niveau de phrases réutilisables et nombre de mots réutilisables), puis sur une métrique automatique (score BLEU).

3.1.6 Évaluations humaines

Pour évaluer les performances de chaque moteur, nous avons confié la relecture des traductions réalisées sur chaque échantillon à des traducteurs/relecteurs humains. Pour ce faire, nous avons utilisé deux méthodes d'évaluation humaine : tout d'abord, le nombre de phrases (segments) nécessitant une réécriture complète par rapport au nombre de phrases total, puis le nombre de mots corrigés par rapport au nombre de mots total, selon quatre catégories : a/ erreurs de sens, b/ erreurs de syntaxe et typos, c/ erreurs de terminologie et d/ erreurs d'entités nommées et de DNT (pour « Do not translate », contenu ne devant pas être traduit, tels que les noms d'entreprise, de marque, etc.). (Voir Annexe IV : Guide d'annotation).

Pour cette deuxième approche, nous nous sommes en partie inspirés du modèle MQM-DQF, que nous avons simplifié, afin d'obtenir des résultats plus génériques pour l'évaluation de la

traduction automatique sur les différents types de texte évalués. Toujours par souci de simplicité, nous n'avons pas jugé nécessaire d'ajouter un critère de sévérité pour ces évaluations.

Nous avons ensuite pondéré les résultats obtenus en rapportant dans un premier temps le nombre d'erreurs trouvées au nombre total de phrases pour le critère de réutilisabilité (1), puis dans un second temps au nombre total de mots pour les catégories restantes (2 à 5), afin de mettre en regard les résultats obtenus.

L'approche visant à évaluer la part de segments « exploitables » est la plus couramment utilisée en pratique, car elle permet d'obtenir rapidement une évaluation proche de la réalité et avec un coût total limité. Il s'agit d'une approche pragmatique, mais souvent subjective, car les résultats obtenus sont souvent liés à la sensibilité du relecteur. L'établissement d'une moyenne entre plusieurs relecteurs permet néanmoins de dégager des tendances et d'établir un « taux de réutilisabilité des segments ».

L'approche par type d'erreurs est plus complexe à mettre en place, mais offre une vision plus fine de la qualité des traductions produites. Des exemples d'un texte avec annotations des types d'erreurs corrigé sont disponibles dans l'Annexe III du présent mémoire.

3.1.6.1 Résultats bruts : Évaluations humaines

Le premier tableau présente les résultats sur le pourcentage de segments réutilisables après évaluation humaine, le second tableau présente le nombre de mots réutilisables après correction par type d'erreurs. Pour des raisons de confidentialité, nous avons anonymisé les références aux clients et aux produits, et nous avons numéroté les outils testés.

	Outil 1	Outil 2	Outil 3	Outil 4	Outil 5	Outil 6	Outil 7
Juridique (licence)	95%	90%	93%	78%	83%	88%	83%
Marketing (présentation)	97%	97%	91%	69%	91%	100%	91%
Marketing (sous-titres)	100%	97%	91%	64%	100%	100%	94%
Chaînes logicielles	80%	90%	86%	50%	76%	74%	92%
Juridique (contrat)	96%	100%	100%	84%	96%	98%	98%
Note moyenne	94%	95%	92%	69%	89%	92%	92%

Tableau 2 : *Tableau récapitulatif, pourcentage de phrases réutilisables par texte après évaluation humaine*
Légende : *(moyenne sur deux annotateurs, rouge si inférieur à 90 %)*

	Outil 1	Outil 2	Outil 3	Outil 4	Outil 5	Outil 6	Outil 7
Juridique (licence)	98%	97%	95%	84%	93%	95%	92%
Marketing (présentation)	98%	95%	90%	59%	91%	93%	90%
Marketing (sous-titres)	97%	94%	93%	66%	90%	91%	93%
Chaînes logicielles	84%	91%	79%	35%	75%	81%	90%
Juridique (contrat)	98%	100%	96%	75%	100%	99%	97%
Note moyenne	95%	95%	91%	64%	90%	92%	92%

Tableau 3 : *Tableau récapitulatif, pourcentage de mots réutilisables par texte après évaluation humaine*
Légende : *(un seul annotateur sur les critères précités, rouge si inférieur à 90 %)*

3.1.6.2 Observations, interprétations et biais potentiels

Les résultats des deux évaluations manuelles indiquent une forte corrélation entre les deux séries de notes pondérées pour chaque type de texte (coefficient de corrélation de 0,99 sur la note moyenne, et entre 0,88 et 0,96 sur les notes des différents échantillons). Ces premiers résultats montrent également une tendance favorable pour les outils 1 et 2, sans différence significative avec les autres outils testés, à l'exception de l'outil 4, dont les résultats sont bien inférieurs. On constate également que les résultats dépendent de la nature du texte, plusieurs moteurs présentant des résultats faibles sur la traduction des chaînes logicielles.

Malgré cette corrélation entre les deux tableaux, on peut toutefois noter plusieurs biais potentiels. Tout d'abord, la taille des échantillons étant faible, il est possible que les résultats ne soient pas suffisamment représentatifs. Ces résultats ne sauraient donc être extrapolés à l'ensemble des textes, et doivent être interprétés comme des tendances. Des tests similaires, effectués sur des échantillons plus gros et avec davantage d'annotateurs, devront donc être mis en œuvre afin de valider ces résultats. De plus, bien que non précisé dans les résultats, on observe à l'issue de la première évaluation un accord inter-annotateur faible, avec cependant une corrélation forte entre les deux résultats finaux moyennés (0,93).

On peut néanmoins supposer que, malgré un accord inter-annotateur faible, un annotateur plus exigeant aura tendance à suivre les mêmes principes pour son annotation et obtiendra donc une moyenne finale corrélée avec celles d'un annotateur globalement moins exigeant. Toutefois, nous ne pouvons prendre cette hypothèse pour acquise et devons mettre en place une seconde salve de test avec des instructions plus précises (par exemple, en définissant le niveau de qualité attendu et en indiquant divers cas de figure), afin d'obtenir un meilleur accord inter-annotateur afin d'augmenter la précision des résultats, et ainsi confirmer ces résultats.

De même, les résultats du deuxième tableau n'étant établis que sur la base d'un seul annotateur, nous ne pouvons considérer ces résultats comme définitifs, et devons réitérer l'expérience sur plusieurs annotateurs, avec le même guide d'annotation.

3.1.7 Évaluation automatique avec BLEU

Afin de vérifier la corrélation avec les métriques automatiques, nous avons comparé les résultats des évaluations humaines précédentes avec ceux de l'évaluation automatique du score BLEU (Kishore Papineni, 2002), largement utilisée pour l'entraînement de moteurs de traduction automatique.

Le score BLEU permet d'évaluer le nombre de tokens consécutifs (n-grams) équivalents entre une phrase traduite par une machine et une traduction de référence, indépendamment de la langue source et cible, avec une pénalité de longueur si le segment candidat est plus court que le segment de référence. (Bonnie Dorr, 2010) La méthode par défaut utilisée par la fonction bleu_score de la bibliothèque NLTK est la méthode BLEU-4, évaluant les suites de 4-grams[8]. (voir Script IV en annexe.)

Contrairement à des métriques plus rudimentaires comme la métrique WER (Word Error Rate), BLEU permet de tenir compte de la séquentialité des tokens et offre donc une précision plus grande. Bien que cette métrique souffre de plusieurs lacunes, notamment en ce qui concerne l'évaluation des variations sémantiques et syntaxiques, liées à la variabilité des traductions possibles, ou l'évaluation des langues morphologiquement riches (Pan, 2016) (Tatman, 2019), elle est aujourd'hui utilisée comme la métrique de référence pour évaluer la qualité des traductions produites en sortie.

3.1.7.1 Résultats bruts : Score BLEU

	Outil 1	Outil 2	Outil 3	Outil 4	Outil 5	Outil 6	Outil 7
Juridique (licence)	29,4	27,2	24,4	13,7	26,1	19,5	25,6
Marketing (présentation)	19,6	20,3	15,4	10,2	15,1	16	16,4
Marketing (sous-titres)	11,7	14,1	12,3	8,4	14,3	10,7	12,1
Chaînes logicielles	21	21	25,3	19,1	47,7	55,8	24,9
Juridique (contrat)	49,9	49,9	42,8	24,7	42,1	39,6	46,1
Note moyenne	26,3	26,5	24	15,2	29	28,3	25

Tableau 4 : *Tableau récapitulatif : scores BLEU par texte*
Légende : *(rouge si inférieur à 25) 0 : complètement différent par rapport à la traduction humaine de référence, 100 : complètement identique à la traduction humaine de référence*

3.1.7.2 Observations, interprétations et biais potentiels

Les résultats de l'analyse avec le score BLEU semblent corroborer en partie ceux de l'évaluation manuelle au niveau de la note pondérée, en attribuant une note légèrement plus faible pour l'outil 3 et une note bien inférieure pour l'outil 4. En revanche, les outils 1 et 2 n'apparaissent plus en tête des meilleurs moteurs. Cette divergence peut être liée aux limites du score BLEU en matière d'analyse syntaxique et sémantique, mais nous ne pouvons pas nous prononcer tant les écarts entre les résultats finaux sont relativement faibles.

[8] https://www.nltk.org/_modules/nltk/translate/bleu_score.html

Dans le détail, on constate que les textes plus rédactionnels (marketing/sous-titres) présentent un score globalement moins élevé, alors que les chaînes logicielles présentent un score globalement plus élevé, ce qui n'était pas le cas dans les évaluations manuelles. Ces deux différences peuvent être liées aux faiblesses du score BLEU, qui aura tendance à attribuer une note plus faible aux contenus présentant une certaine « malléabilité » syntaxique et sémantique (typique des textes à caractère marketing), et à privilégier les textes plus suivant une syntaxe et un vocabulaire plus « contrôlé » (typique des textes techniques, mais également juridiques et informatiques).

Afin de pallier ces biais et ainsi obtenir des résultats plus significatifs, nous pourrions, comme dans l'évaluation manuelle, tester nos outils sur des échantillons plus variés et plus volumineux, voire à l'aide d'autres métriques. Ce changement permettrait notamment de lisser les résultats globaux. Encore une fois, les résultats obtenus ne sont que des indicateurs sur la qualité générale des textes produits et une seconde salve de tests sur d'autres échantillons représentatifs serait utile afin de départager les différents outils, tout en tenant compte des biais induits par le score BLEU.

3.1.8 Conclusion de l'étude 1 et pistes d'amélioration

Dans cette première étude, nous avons testé plusieurs outils NMT disponibles sur le marché pour la paire de langue EN>FR à l'aide d'une série d'échantillons représentatifs des divers types de textes traduits au sein d'Univoice et de critères d'évaluation humains et automatiques.

Les premiers résultats nous ont permis de dégager une tendance sur la qualité générale des traductions produites par les différents outils, mais les résultats restant relativement proches, nous ne pouvons pas nous en contenter pour le choix d'un outil en particulier, notamment en raison de la taille réduite des échantillons testés. D'autres tests menés sur des échantillons plus larges, avec davantage d'annotateurs et des critères plus précis peuvent donc être mis en place pour départager de manière plus catégorique les différents outils.

Bien que l'évaluation humaine soit plus proche de la réalité, cette dernière ne peut malheureusement pas être intégrée à une chaîne de traitement pour l'entraînement de moteurs de traduction automatique. Compte tenu des disparités entre l'évaluation humaine, plus fiable pour déterminer l'effort à fournir en post-édition, et l'évaluation automatique BLEU, nous devrons donc prendre les résultats de cette dernière métrique avec attention et tenir compte des biais induits.

La possibilité d'utiliser d'autres métriques automatiques, ou d'une moyenne pondérée de plusieurs métriques, devra être considérée pour de nouvelles évaluations, et l'entraînement de moteurs de traduction. Nous devrons également tenir compte des faiblesses de chaque métrique, en considérant notamment la longueur moyenne des phrases testées (notamment pour les chaînes logicielles) et le type de texte évalué (marketing, rédactionnel) pour interpréter les résultats d'évaluation.

Bien qu'insuffisants, ces premiers résultats nous ont malgré tout servis de point de départ pour réduire le panel d'outils testés et limiter ainsi l'effort d'évaluation à fournir. Étant contraint de choisir parmi ce panel, notre choix s'est arrêté sur l'outil 2, en raison de ses performances légèrement supérieures, mais également de son coût et de sa simplicité d'intégration à la plateforme.

3.2 Étude 2 : étude de faisabilité sur l'entraînement de moteurs de traduction automatique neuronale avec OpenNMT

Nous décrirons dans cette partie le processus d'entraînement de moteurs de traduction automatique neuronale en interne à l'aide de la technologie OpenNMT.

Projet	**Entraînement de moteur de traduction automatique neuronale en interne**
Objectifs	-> *Entraîner des moteurs de traduction automatique neuronale afin de tester la possibilité d'une utilisation de la technologie OpenNMT en interne*
Équipe	- 1 taliste (mise en place de l'entraînement et des tests) - 1 développeur back-end (gestion des serveurs et des containers) - 1 traducteur junior, 1 traducteur/relecteur senior (évaluation humaine)
Technologie	- TAL : Python, script bash, OpenNMT (Torch, Lua)
Tâches	- Prendre en main la technologie OpenNMT - Collecter des corpus bilingues alignés - Entraîner des moteurs test et évaluer les résultats obtenus

3.2.1 Objectifs et étapes préliminaires

L'objectif général de cette étude est d'évaluer la possibilité d'intégrer un moteur de traduction automatique pour la combinaison EN > FR à la plateforme pour différents domaines de spécialité, en vue d'une exploitation ultérieure en production. L'étude s'appuie sur la technologie OpenNMT, largement utilisée par les moteurs NMT actuels, tels que Google NMT ou Systran, et permettant l'entraînement de modèles de traduction automatique neuronale.

L'étape d'entraînement nécessitant de grandes quantités de données, nous avons dû récupérer des mémoires de traduction par paire de langues/domaine libres de droit. Le site OPUS[9] regroupe une large collection de corpus multilingues alignés disponibles en open source, composés de traductions humaines en provenance de sources diverses (logiciels open source, sites de l'Union européenne, pages Wikipédia, entre autres).

Nom de la mémoire de traduction	**Type/domaine**	**Nombre de segm. alignés**	**Nombre de mots total**
GlobalVoices	éditorial	339K	13,6M
OpenSubtitle2018	sous-titres (divers)	34,4M	496M
TedTalks2013	sous-titres (conférences)	159K	5,46M
EMEA - Europ. Med. Ag.	médical	373K	11,58M
EU bookshop	générique, rapports officiels	9,4M	449M
Gigaword	générique, éditorial	21,9M	1,21G
KDE4, GNOME, Ubuntu	interface, chaînes logicielles	244,4K	4,52M
Wikipedia	générique, encyclopédique	818K	33,9M

Tableau 5 : *Tableau des corpus collectés sur le site d'OPUS*

[9] http://opus.nlpl.eu/

3.2.2 Fonctionnement général de la NMT et d'OpenNMT

Comme nous l'avons vu dans la partie présentant les concepts clés, la technologie de traduction neuronale est à ce jour l'approche présentant les meilleures performances. (Quoc V. Le, 2016). Contrairement aux approches déterministes basées sur des règles développées depuis les 50 (RBMT, Rule-Based Machine Translation), elle s'appuie sur une approche statistique, plus pertinente pour prendre en compte la variabilité de la langue. (Systran Blog, 2016) À la différence de l'approche statistique traditionnelle, développée depuis les travaux d'IBM dans les années 90, l'approche neuronale (NMT) tire parti, comme son nom l'indique, des technologies de réseaux de neurones et d'apprentissage profond. (Aaron Han, 2016)

Les systèmes NMT sont entraînés sur la base de corpus bilingues alignés (ou parallèles), où chaque segment est représenté sous forme de vecteurs selon un modèle « seq2seq » (séquence à séquence), modélisés à l'aide d'un mécanisme d'encodeur-décodeur (Dugar, 2019). Cette méthode est renforcée par un mécanisme d'attention permettant de traiter des séquences de mots plus longues. (Lamba, 2019). L'objectif de cette méthode est de modéliser la relation qu'entretient un segment dans une langue source et une langue cible, afin de créer in fine un modèle permettant de générer des traductions candidates.

Les systèmes de traduction automatique neuronale s'appuient sur des réseaux de neurones dits « récurrents » (RNN : Recurrent Neural Network) dans la mesure où l'apprentissage machine s'effectue sur plusieurs éléments d'une même séquence de manière cyclique, et plus particulièrement sur des réseaux dits LSTM (Long-Short Term Memory) (Olah, 2015) permettant de conserver les informations d'un token à l'autre à l'échelle d'un segment. Les modèles de traduction générés à l'aide d'un algorithme d'apprentissage à base de réseaux de neurones sont appelés « moteurs NMT » et peuplent aujourd'hui le paysage de la traduction automatique avec des moteurs tels que GNMT de Google ou encore le moteur Pure NMT développé par Systran.

Malgré des résultats probants, l'entraînement de ces modèles reste complexe, notamment en raison de la taille des corpus et du temps d'entraînement nécessaires pour l'obtention de résultats significatifs. De même, comme toutes les technologies non déterministes d'apprentissage automatique, le modèle une fois entraîné ne peut être corrigé ponctuellement, et nécessite un réentraînement complet, où seuls les paramètres et les corpus en entrée peuvent éventuellement être ajustés, jusqu'à obtention d'un moteur performant.

Développée depuis 2016 par le groupe Harvard NLP, la technologie OpenNMT[10] (Guillaume Klein, 2018) permet l'entraînement et le déploiement de moteurs de traduction automatique neuronale. Le kit offre une série de fonctionnalités uniques et un paramétrage fin pour l'apprentissage seq2seq et la création de modèles de traduction automatique, mais aussi pour la reconnaissance vocale ou la lecture d'images. L'écosystème s'appuie notamment sur la bibliothèque d'apprentissage automatique Pytorch, permettant la manipulation de tenseurs et de gradients. (Voir Annexe VI : Écosystème OpenNMT)

3.2.3 Méthodologie

Pour cette étude de faisabilité, nous avons voulu tester le fonctionnement de base d'entraînement et de prédiction du kit OpenNMT sur un premier corpus spécialisé, afin de tester ses

[10] http://opennmt.net/

capacités et d'évaluer la qualité des premiers résultats en sortie. L'entraînement se découpe en trois phases : une première phase de prétraitement du corpus d'entraînement (preprocess.lua, utilisé notamment pour la tokenisation), l'entraînement du modèle à proprement parler (train.lua), et enfin la phase de prédiction sur un jeu de données test issu du corpus (translation.lua). Nous avons automatisé ces trois étapes à l'aide d'un script, en récupérant, à l'issue de la phase d'entraînement, l'epoch ayant produit les meilleurs résultats, autrement dit le niveau de perplexité (PPL) le plus faible.[11] (Voir Script I et II en annexe.)

Pour notre expérience, nous nous sommes concentrés pour l'heure sur un premier jeu de données spécialisé dans le domaine médical pour la paire de langues EN>FR (Corpus EMEA), que nous avons réparti en trois sous-jeux de données pour l'entraînement : train, dev et test (80-10-10). Nous réalisons cette division à partir des corpus au format TMX converti en bitextes (Voir Script III). L'outil translation.lua d'origine nous permet d'obtenir le score BLEU et TER global. Nous consignons l'ensemble des traitements dans un log afin de conserver une trace des opérations et des résultats intermédiaires.

Une fois les prédictions effectuées, nous soumettons les traductions candidates à deux traducteurs/relecteurs, comme dans notre première étude comparative, afin que ces derniers évaluent les 200 premiers segments sur la base du critère de réutilisabilité et établissent un bilan des erreurs résiduelles. Nous consignons les données de test, afin de pouvoir ajuster les paramètres et lancer une nouvelle salve d'entraînement de manière itérative. Nous pouvons ainsi entraîner plusieurs modèles sur la base d'un même jeu de tests à l'aide de paramètres différents en partant des résultats de la première salve comme baseline.

Nom du modèle	**Demo-emea (default parameters)**
Corpus	EMEA (open-source), source : OPUS
Description du corpus	This is a parallel corpus made out of PDF documents from the European Medicines Agency. All files are automatically converted from PDF to plain text using pdftotext with the command line arguments -layout -nopgbrk -eol unix. There are some known problems with tables and multi-column layouts - some of them are fixed in the current version.
Date	17/05/2019
Nb seg. total	373 152 segments
Nb seg./mots train	298 521 segments / 4 335 621 mots (EN)/ 4 960 889 mots (FR)
Nb seg./mots dev	37 316 segments / 516 621 mots (EN)/ 586 922 mots (FR)
Nb seg./mots test	37 315 segments / 468 156 mots (EN)/ 531 001 mots (FR)
Meilleure epoch/lowest PPL	Epoch_9_11.0
Temps d'entraînement	17.05 15:37 > 20.05 4:25 (2j 14h = 62h pour 9 epochs)

Tableau 6 : *Tableau récapitulatif du modèle entraîné (Demo-emea)*

[11] https://en.wikipedia.org/wiki/Perplexity

Les scripts d'entraînements et de génération des traductions sont lancés depuis un container Docker géré par un développeur back-end afin de ne pas encombrer les serveurs. Compte tenu de la charge que représente l'entraînement, nous ne pouvons lancer qu'un seul entraînement de modèle à la fois ou une seule salve de génération de traductions. Comme on le constate dans le précédent récapitulatif, le temps nécessaire à l'entraînement d'un modèle sur un corpus d'environ 4,3 millions de mots est de l'ordre de 2,5 jours sur un processeur unique à raison de 9 epochs.

3.2.4 Premiers résultats

Après entraînement du modèle Demo-emea sur le corpus EMEA et génération des traductions candidates sur le corpus test, nous obtenons les résultats suivants pour le paramétrage de base :

Expérience	**Baseline**
Ligne de commande	th translate.lua -model data/demo-emea/en-fr-emea.tmx/demo-emea-model_epoch9_11.00.t7 -src data/demo-emea/en-fr-emea.tmx/test_en.txt -output data/demo-emea/en-fr-emea.tmx/pred_2.txt -time true -replace_unk true
Paramètres	Default, un seul CPU, paramètre replace_unk = true
Temps de traduction	12:24 > 14:02 (1h34)
# <unk> src	47 002 (EN), 0 (FR)
Score BLEU	22.49 BLEU = 22.49, 52.6/33.3/23.0/16.3 (BP=0.791, ratio=0.810, hyp_len=439485, ref_len=542613)
Score TER	69.71 TER = 69.71 (Ins 1.4, Del 10.9, Sub 8.0, Shft 1.1, WdSh 1.5)
Éval. humaine, réutilisabilité	Environ 70,88 % de segments réutilisables (moyenne 2 annotateurs)

Tableau 7 : *Tableau récapitulatif des résultats obtenus pour la baseline*

Après génération, nous relevons les scores BLEU et TER calculés automatiquement sur le jeu de données test, et établissons une moyenne inter-annotateur des deux traducteurs/relecteurs, afin d'obtenir un score de réutilisabilité global (dans notre cas, 70,88 %). Ces premiers résultats affichent un score globalement plus faible que ceux obtenus lors de l'évaluation des outils du marché. Ces résultats ne sont pas surprenants étant donné qu'il s'agit d'un premier entraînement.

Malgré ce score plus faible, on constate que les traductions obtenues en sortie à l'issue cette première salve d'entraînement ne sont pas inexploitables et peuvent être améliorés. On constatera notamment dans le bilan des traducteurs/relecteurs divers problèmes en sortie que nous tenterons d'éliminer sur les prochains entraînements.

#	Problème	Exemple	Solution potentielle
1	Espaces après les apostrophes	si l' un des symptômes énumérés ci-dessus apparaît	Post-traitement
2	segments difficilement traités par le moteur en raison de leur taille ou de leur segmentation	Common: • lethargy • dizziness • shakiness or tremor • nausea • diarrhoea • vomiting • rash or skin eruptions (exanthema) • pain in your joints (arthralgia) or muscles (myalgia) • back pain • feeling dizzy or faint when you stand up suddenly (orthostatic hypotension) • swelling (typically in ankles or feet) caused by fluid retention (oedema) • tiredness • vivid dreams • confusion • feeling anxious • sleeping problems	Nettoyage du corpus : Élimination des segments trop longs ou non conformes des corpus (lié potentiellement au format d'origine du TMX, à savoir des documents PDF) + nouvelle phase d'entraînement
3	mots non traduits et laissés en EN dans la cible, même des mots basiques	même lorsque votre dépression a lifted. / name et address / nozzle / picture / liquid / tighten / metering / salt / tolerated /days / rarely / with / skip	Prétraitement : Modifier la tokenisation et la normaliser le texte (notamment la casse) afin de limiter les mots out-of-vocabulary (OOV : termes absents du corpus d'entraînement et présents dans le corpus de test). Utilisation d'un dictionnaire avec l'option –phrase_ table
4	number mismatches	105 PACKAGE LEAFLET: -> 110 NOTICE:	Utilisation de placeholders pour les nombres, les URL et les dates[12]
5	segment « non traduit »	Reproduction is authorised provided the source is acknowledged.	Nettoyage du corpus : La phrase apparaît telle que dans le corpus cible d'entraînement. Nettoyage du corpus requis si contenu laissé en anglais.
6	Incohérences termino	Des mots clés fréquents sont parfois traduits de 2 façons différentes (« pump » traduit soit par « seringue », soit par « pompe » par exemple)	Difficile à corriger, erreur intrinsèque au fonctionnement de la NMT

Tableau 8 : *Tableau récapitulatif des erreurs relevées par les annotateurs et solutions*

Comme on le constate dans ce premier récapitulatif des commentaires des annotateurs, les traductions présentent des erreurs variées pouvant être traitées séparément, que ce soit au niveau du prétraitement (normalisation, tokenisation), à l'échelle du corpus (avec l'élimination des segments trop longs, mal segmentés ou laissés en anglais), ou en post-traitement.

En plus de ces problèmes ponctuels résolubles, on sait d'après les précédentes études menées que la qualité des résultats obtenus avec les moteurs NMT est fortement corrélée à la taille du corpus en entrée, nous pouvons donc tenter de jouer sur la taille de ce dernier afin d'obtenir de meilleurs résultats, en le complétant par exemple avec d'autre corpus du même domaine (corpus

[12] http://forum.opennmt.net/t/handle-numbers-urls-dates/2307/2

enrichi), en le fusionnant à des corpus plus génériques (corpus mixte) ou à l'aide de plusieurs salves d'entraînement (adaptation au domaine). Nous pouvons également faire varier les paramétrages, notamment le nombre d'epochs ou le recours au paramètre BPE (Byte Pair Encoding), plus efficace pour la prédiction des termes « out-of-vocabulary » (Rico Sennrich, 2016), et analyser les résultats produits.

3.2.5 Conclusion de l'étude 2 et pistes d'amélioration

Dans cette étude de faisabilité, nous avons souhaité déterminer la viabilité d'un éventuel développement de moteurs NMT en interne grâce à la technologie OpenNMT. Nous avons pris comme point de départ le corpus EMEA (médical) que nous avons scindé en sous-corpus et utilisé pour créer un modèle de prédiction. Après évaluation des premières traductions en sortie, il semble que les textes traduits présentent une qualité, certes moins élevée que les outils génériques disponibles sur le marché, mais néanmoins suffisamment bons pour être exploités et améliorés.

Nous avons relevé plusieurs erreurs ponctuelles que nous pourrons corriger sur de prochaines salves d'entraînement. Nous pourrons également tester d'autres configurations afin de procéder à de nouvelles évaluations et déterminer ainsi la meilleure configuration possible pour l'entraînement.

Malgré ces résultats satisfaisants, nous continuons de nous heurter à plusieurs problèmes. Tout d'abord, certaines erreurs, notamment en matière terminologique sont difficilement résolubles compte tenu des propriétés intrinsèques des moteurs NMT. De plus, nous nous heurtons au problème de taille du temps et de la capacité nécessaires à l'entraînement d'un modèle sur la base d'un corpus (plus de deux jours pour une seule salve d'entraînement).

Pour pouvoir tirer pleinement parti de cette technologie et déployer une batterie de tests suffisants, qui plus est dans plusieurs domaines spécialisés et dans plusieurs paires de langues, nous devrons impérativement utiliser un parc de serveurs plus étendu et plus performant (par exemple, le recours à des GPU en lieu et place des CPU actuels). Le critère d'investissement en temps et en matériel est donc un facteur décisif pour mesurer la viabilité de telles recherches à long terme.

Conclusion générale

Dans ce mémoire, nous nous sommes intéressés aux problématiques d'intégration de la technologie de traduction automatique neuronale au sein d'une agence de traduction. À cette fin, nous avons étudié la mise en place et les résultats de deux études de cas, une première visant à évaluer la qualité de plusieurs outils NMT actuellement disponibles sur le marché à l'aide de métriques d'évaluation, à des fins de partenariat futur, et une seconde, visant à déterminer la viabilité d'un éventuel développement de moteurs NMT en interne grâce à la technologie OpenNMT.

Ces deux études s'inscrivent dans le cadre du développement d'une plateforme marketplace en interne, via une équipe de développeurs dédiée, destinée à développer une clientèle directe et de s'affranchir des grandes agences de traduction multilingue et de leur mainmise technologique. Une grande partie du développement de cette plateforme repose sur le traitement de documents

multilingues, et l'intégration de la technologie de traduction automatique neuronale, que ce soit via des fournisseurs externes ou le développement de moteurs de traduction spécialisés en interne.

À l'issue de la première étude visant à analyser les différents outils NMT disponibles sur le marché sur la base de la qualité des traductions produites, nous sommes parvenus à mettre en avant un moteur légèrement plus performant, malgré les faiblesses de notre première salve d'évaluation, et à mettre en évidence les divergences et convergences entre les différentes métriques. Une évaluation tirant parti de ces premiers résultats peut néanmoins s'avérer nécessaire, en sus de critères purement marketing et techniques.

La seconde étude visant à tester la viabilité de l'entraînement de moteurs de traduction automatique neuronale en interne à l'aide de la technologie OpenNMT nous a permis quant à elle de constater la qualité des traductions produites à l'issue d'un premier entraînement test sur un corpus de données spécialisé (dans notre cas, le domaine médical). Nous avons également pu relever les faiblesses de ce premier moteur pour un éventuel réentraînement ultérieur. Toutefois, malgré ces résultats positifs, nous constatons que l'implémentation de cette technologie, en particulier l'entraînement de modèles, représenten un coût matériel et un temps de développement non négligeables. Le choix du développement de moteurs en interne devra donc être étudié d'un point de vue stratégique, au vu de sa rentabilité par rapport aux moteurs de traduction génériques disponibles sur le marché.

Bibliographie

Aaron Han, D. W. (2016). *Machine Translation Evaluation: A Survey.*

Bonnie Dorr, M. S. (2010). *Part 5: Machine Translation Evaluation.*

Christophe Servan, J. C. (2016). *Domain specialization: a post-training domain adaptation for Neural Machine Translation.*

Dugar, P. (2019, Juillet). *Attention — Seq2Seq Models*. Retrieved from Towards Data Science: https://towardsdatascience.com/day-1-2-attention-seq2seq-models-65df3f49e263

German Research Center for Artificial Intelligence (DFKI) and QTLaunchPad. (2015, Juin). *Multidimensional Quality Metrics (MQM) Definition*. Retrieved from http://www.qt21.eu/mqm-definition/definition-2015-12-30.html

Guillaume Klein, Y. K. (2018). *OpenNMT: Neural Machine Translation Toolkit.*

Kishore Papineni, S. R.-J. (2002). *BLEU: a MethodforAutomaticEvaluationofMachineTranslation.*

Lamba, H. (2019, Mars). *Intuitive Understanding of Attention Mechanism in Deep Learning*. Retrieved from Towards Data Science: https://towardsdatascience.com/intuitive-understanding-of-attention-mechanism-in-deep-learning-6c9482aecf4f

Olah, C. (2015, Août). *Understanding LSTM Networks*. Retrieved from Colah's Blog: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Open Standards for Container/Content Allowing Re-use (LISA). (1997). *TMX format Specifications*. Retrieved from http://xml.coverpages.org/tmxSpec971212.html

Organization for the Advancement of Structured Information Standards XLIFF TC. (2008). *XLIFF format Specifications*. Retrieved from https://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html

Pan, H. M. (2016, Novembre). *How BLEU Measures Translation and Why It Matters*. Retrieved from Slator: https://slator.com/technology/how-bleu-measures-translation-and-why-it-matters/

Quoc V. Le, M. S. (2016, Septembre). *A Neural Network for Machine Translation, at Production Scale*. Retrieved from Google AI Blog: https://ai.googleblog.com/2016/09/a-neural-network-for-machine.html

Rico Sennrich, B. H. (2016). *Neural Machine Translation of Rare Words with Subword Units.*

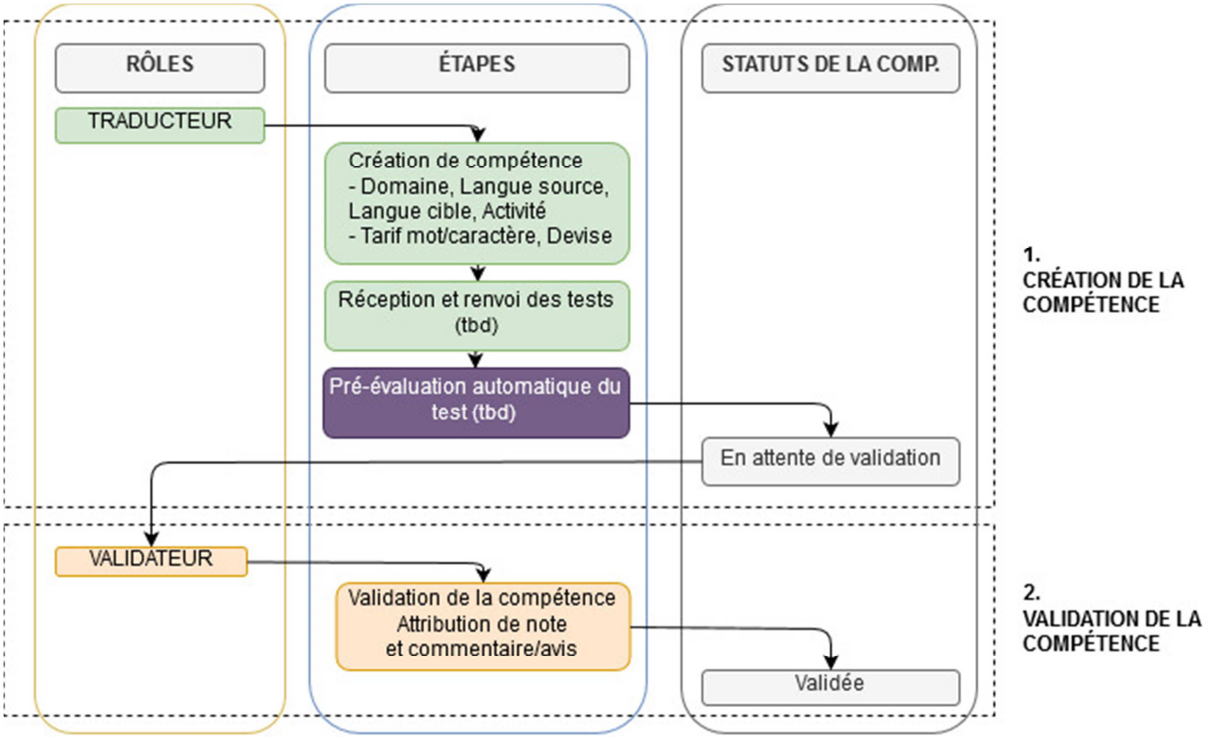
Risku, H. (2007). The role of technology in translation management. In M. S. Yves Gambier, *Doubts and Directions in Translation Studies: Selected contributions from the EST Congress, Lisbon 2004*.

Systran Blog. (2016, Octobre). *How does Neural Machine Translation work?* Retrieved from Systran Blog: https://blog.systransoft.com/how-does-neural-machine-translation-work/

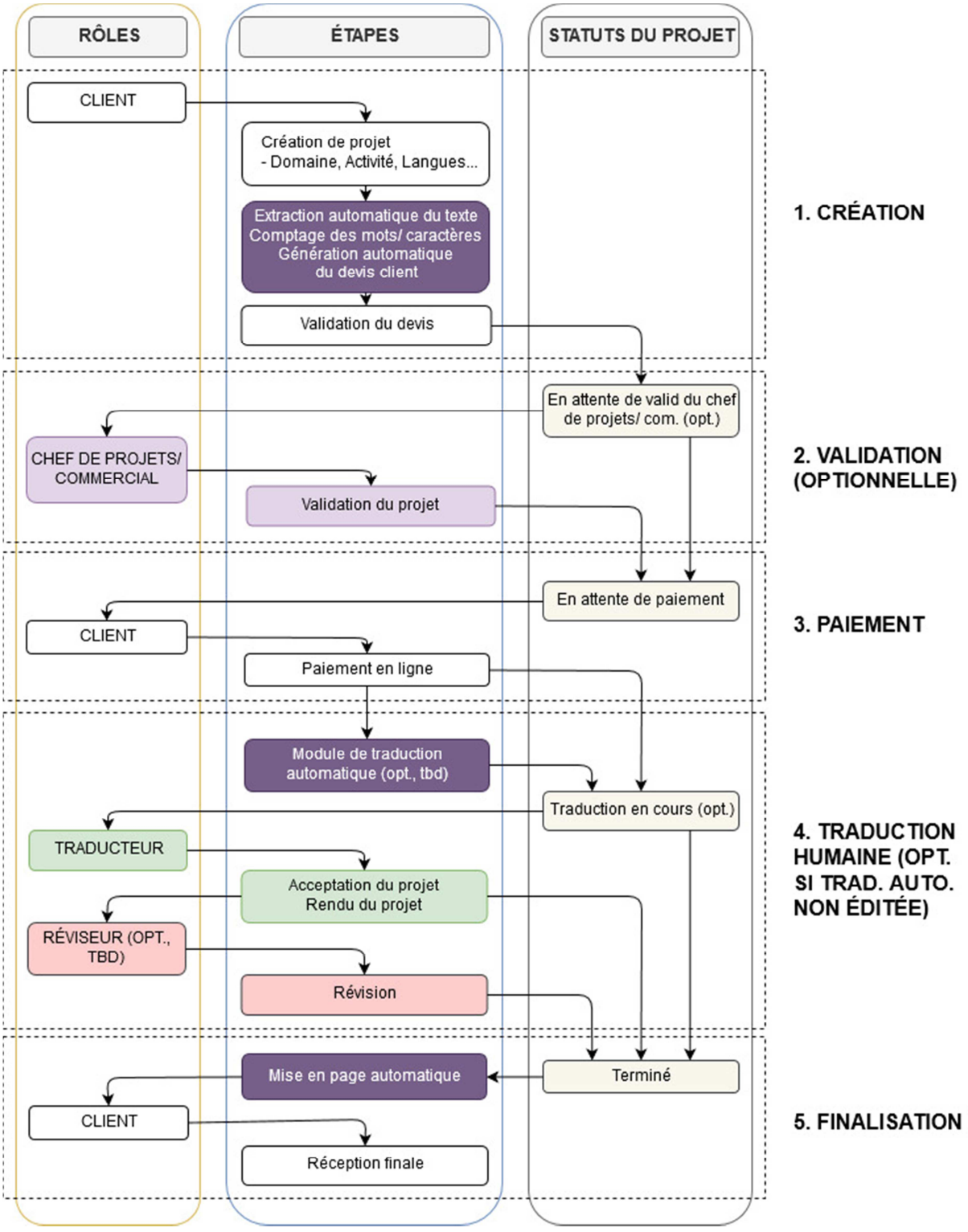
Tatman, R. (2019, Janvier). *Evaluating Text Output in NLP: BLEU at your own risk*. Retrieved from Towards Data Science: https://towardsdatascience.com/evaluating-text-output-in-nlp-bleu-at-your-own-risk-e8609665a213

Annexes

I. Workflow de validation des compétences



II. Workflow de cycle de vie d'un projet



Note : Les étapes impliquant un traitement TAL sont indiquées en mauve.

III. Exemple de texte annoté manuellement avec cinq critères

Exemple de paragraphe annoté avec indication des différents types d'erreur après évaluation d'une traduction automatique (fusion des annotations au niveau du segment : réutilisabilité, et des annotations au niveau du mot : quatre critères d'évaluation basés sur le standard MQM-DQF).

Catégories : **Réutilisabilité** – **Sens** – **Syntaxe et typo** – **Terminologie** – **DNT et entités nommées**

XXX Annonce YYY 5.0, la Dernière Version de **son**
Solution de **CAD** primée

Le lancement présente des capacités **de percée** dans la fabrication additive, l'optimisation de **Topology** et la simulation

NEEDHAM, Massachusetts. – Le 19 mars 2018 – XXX (NASDAQ : XXX) **aujourd'hui a annoncé** YYY 5.0, la dernière **libération** de son logiciel de conception assistée par ordinateur (**CAD**) de YYY® 3D, qui permet aux utilisateurs d'aller du concept à la fabrication dans un environnement de **design.** YYY 5.0 introduit cinq nouvelles **capacités du monde changeant vite de design** de produit et présente des améliorations de productivité clés.

YYY permet aux compagnies d'accélérer **l'innovation de produit** et construire de meilleurs produits plus vite en réutilisant les meilleurs designs et **en remplaçant des hypothèses avec les faits**. **Avec YYY 5.0, les concepts peuvent être transformés dans les produits intelligents, connectés, en construisant un pont sur les mondes physiques et numériques avec les capacités de réalité augmentée (AR) dans chaque siège**. XXX 5.0 introduit aussi des capacités **excitantes** dans les régions d'optimisation **topology,** fabrication additive et soustractive, **dynamique liquide quantificative** et CAM.

"**XXX est sur le principal bord de certaines des technologies les plus chaudes aujourd'hui avec Internet of Things (IoT) et réalité augmentée (AR), mais il n'a pas oublié ses racines dans la CAD, en transformant plutôt ces affaires en insufflant son principal logiciel YYY à de nouvelles technologies et aux capacités**", ont dit XXX, le président, YYY. "YYY 5.0 est encore un exemple de l'innovation **continuée** de XXX et de la capacité de **rencontrer** des besoins de client."

Optimisation de **Topology**

Le design physique **de produits** est souvent limité par les designs existants et les pratiques. La nouvelle **Extension d'Optimisation de YYY Topo** crée automatiquement des designs optimisés basés sur un **ensemble défini des objectifs** et des contraintes, **relâchées** par les designs existants et **croyait des processus**. Cela aide **des utilisateurs** à **économiser le temps** et accélérer l'innovation en permettant la création de **parties** optimisées et efficaces.

IV. Guide d'annotation : Définition des critères d'évaluation

Réécriture : Cette catégorie vise à évaluer les phrases dans leur globalité et non les problèmes de traduction internes à la phrase. Il permet de déterminer si une phrase prétraduite par un moteur de traduction automatique peut être corrigée ponctuellement ou doit être totalement réécrite par le post-éditeur, autrement dit s'il est plus rapide pour un post-éditeur de retraduire la phrase en partant de zéro plutôt que d'utiliser la phrase proposée par le moteur. Cette mesure est particulièrement importante pour juger de la qualité d'une traduction automatique, car elle permet d'évaluer rapidement le temps nécessaire à la post-édition du document.

Exemple devant être entièrement réécrite :

EN : *We unlock the power of technology by connecting its complex components, so our customers can create better experiences for people.*

FR : *Nous ouvrons le pouvoir de technologie en raccordant ses composantes complexes, donc nos clients peuvent créer de meilleures expériences pour les gens.*

Sens : Correspond à la catégorie d'erreurs « Accuracy » du modèle MQM-DQF, cette catégorie regroupe toutes les erreurs de sens typiques telles que les omissions et additions d'éléments, ainsi que les problèmes de traduction erronée et d'expressions « sous-traduites » (perte de sens) ou « surtraduites » (ajout de sens), voire non traduite, et plus généralement toutes les traductions qui ne transmettent pas le sens du texte d'origine. Cette catégorie d'erreur est la plus problématique, car elle demande une édition et une attention plus importante de la part du post-éditeur.

Exemple :

EN : *Launch Features Breakthrough Capabilities in Additive Manufacturing, Topology Optimization, and Simulation*

FR : *Lancement des fonctionnalités de rupture dans la fabrication d'additifs, l'optimisation de topologie et la simulation.*

Syntaxe et typo : Correspond à la catégorie d'erreurs « Fluency » et « Style » du modèle MQM-DQF, cette catégorie comprend toutes les erreurs relatives à la syntaxe telles que les fautes de grammaire, d'accord, etc. ainsi que les fautes d'orthographe, et les tournures et formulations extrêmement maladroites et non idiomatiques obligeant le post-éditeur à effectuer une reformulation. Les erreurs de syntaxe et d'orthographe sont généralement plus faciles à repérer et à analyser par le post-éditeur, car il ne nécessite presque pas de retour sur le texte source. Les erreurs de tournure et de formulation sont également simples à détecter et à corriger, mais peuvent nécessiter de revenir sur le texte d'origine.

Exemple :

EN : *[…] you are not permitted to install, access or use the software or any part thereof, and within five days after receipt of the software you must return the software to the place where you*

obtained it, or if supplied electronically you must certify destruction of all electronic copies of the software, for a full refund of any money paid.

FR : *[…] on ne vous autorise pas à installer, accéder ou utiliser le logiciel ou toute partie de cela et dans les cinq jours après le reçu du logiciel vous devez rendre le logiciel à l'endroit où vous l'avez obtenu, ou si fourni électroniquement vous devez certifier la destruction de toutes les copies électroniques du logiciel, pour un plein remboursement de tout argent payé.*

Terminologie : Correspond à la catégorie d'erreurs « Terminology » du modèle MQM-DQF, elle comprend les erreurs relatives à la terminologie, telles que les incohérences ou le non-respect du domaine, entre autres. Elle ne prend pas en compte la terminologie client. Cette catégorie d'erreur est particulièrement critique dans les textes techniques comportant des nombreux termes. Nous ajoutons à cette catégorie les incohérences entre les éléments de la traduction des options dans l'interface utilisateur et la documentation. Ce type de catégorie d'erreurs est difficile à détecter, car elle nécessite une bonne connaissance du domaine et une grande vigilance de la part du post-éditeur.

Exemple :

EN : *XXX also introduces exciting capabilities in the areas of topology optimization, additive and subtractive manufacturing, computational fluid dynamics, and CAM.*

FR : *XXX introduit également des capacités passionnantes dans les domaines de l'optimisation topologique, de la fabrication additive et soustractive, de la dynamique des fluides computationnelle et de la FAO.*

DNT et entités nommées : Inclut la catégorie d'erreurs « Locale convention » du modèle MQM-DQF, ainsi que la traduction des termes ne devant pas être traduits dans la langue cible, tels que les noms de produit, et plus généralement les problèmes de traduction relatifs aux noms propres. Cette catégorie est importante, car elle permet de constater l'aptitude du moteur à distinguer les entités nommées et les expressions non traduisibles.

Exemple :

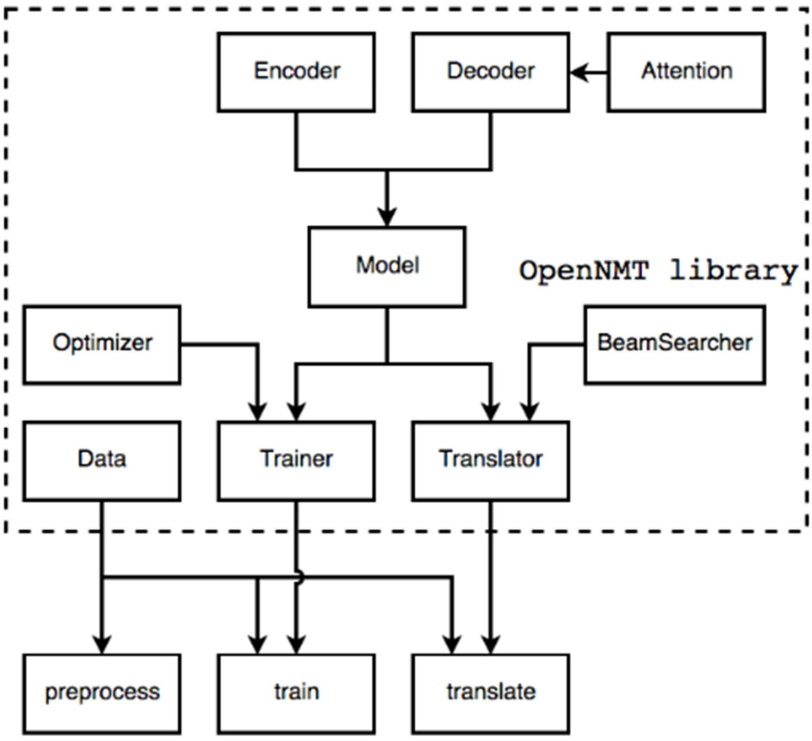
EN : *XXX that has licensed RS Components to promote, sell or otherwise distribute the Product.*

FR : *XXX qui a permis des composants de RS pour promouvoir, vendre ou autrement distribuer le produit.*

V. Récapitulatif des métriques d'évaluation pour la traduction automatique les plus courantes

Nom	Nom complet	L-dep.	Fonctionnement	Caractéristiques
WER	Word Error Rate	Non	Basée sur la distance de Levenshtein. Somme des (S)ubstitutions, (I)nsertions, et Suppressions(D), divisée par le (N)ombre de mots du segment de référence	Difficulté à tenir compte de la variabilité syntaxique
BLEU	BiLingual Evualation Understudy	Non	Calcul du nombre de n-grams (ou séquences de mots, par défaut 4-gram) identiques entre réf. et cand. + pénalité en cas de différence de longueur réf./cand.	Exploitation de références multiples, mesure de la précision et non du rappel
NIST	National Institute of Standards and Technology	Non	Variante de la métrique BLEU, utilisant une méthode plus fine pour le compte de n-grams, avec instauration de poids selon le niveau d'information des tokens	Amélioration de la métrique BLEU avec instauration de poids
METEOR	Metric for Evaluation of Translation with Explicit Ordering	Oui	Moyenne harmonique du nb d'unigrammes équivalents/total du segment candidat –et- du segment de réf.. Trois critères d'équivalence : correspondances exactes, tokens avec racine commune, synonymes (basés sur Wordnet)	Calcul de la moyenne harmonique, recours à Wordnet pour les synonymes
GTM	General Text Matcher	Non	Taille de la plus longue série de tokens similaires candidat/référence divisé par la taille du segment candidat (précision) et réf. (rappel)	Exploitation de références multiples, calcul de la moyenne harmonique
TER	Translation Error Rate	Non	Version étendue de la métrique WER, prenant en compte les déplacements (shift) de tokens (variabilité syntaxique)	Métrique WER + shift
HTER	Human-mediated Translation Error Rate	Oui	Métrique TER où la référence utilisée est une correction manuelle du segment candidat avec un minimum d'édition.	Métrique TER + segment de réf. minimal. Automatisation impossible

VI. Schéma de l'écosystème d'OpenNMT



Note : Schéma issu du document « OpenNMT: Neural Machine Translation Toolkit » (Guillaume Klein, 2018)

Scripts

Script I : Prétraitement, entraînement et évaluation (script.sh)

```
# ------------------------------------------------------------------------
-------------------------------------------------------
# Utility: preprocess training set, train MT engine and evaluate MT output
# 1/ Launch preprocess.lua to preprocess train, dev and test sub-corpora as
created with convert_and_split.py program
# 2/ Launch train.lua to train Machine Translation engine with dir files
# 3/ Determine best epoch using best_epoch.py and remove other epoch files
# 4/ Launch translate.lua to generate MT output on test set
# 5/ Evaluate MT output using tools/score.lua (both BLEU and TER scores)
# ------------------------------------------------------------------------
-------------------------------------------------------
# Usage (from dev server, using open-nmt container):
# sudo docker run -d --name <container_name, ex: train-eval-emea-500> -v
/data/ssh-dev/home/<user_name, ex: victorien>/data:/root/opennmt/data
registry.univoice.fr/opennmt bash -c 'cd data/utils && sh script.sh
<dir_name_in_expe ex: demo-emea-500> | tee -a <path_to_dir, ex:../tms/demo-
emea-500/train-eval.log'>
# ------------------------------------------------------------------------
-------------------------------------------------------
# Note: Languages still hard coded for now (en-fr), to be included as
parameters/retrieved on the fly. Training time depends on
# number of segments in training set, number of cpus used, total number of
epochs (still very long to obtain reasonable scores)
# ------------------------------------------------------------------------
-------------------------------------------------------
# INPUT: directory containing train, dev and test subcorpora (in expe
directory by default), directory name should reflect the
# exeperience you want to make
# OUTPUT: 2 dict files in both en and fr language, one trained model (best
epoch), one pred.txt file containg MT output,
# one log file as stated in tee command (optional)
# ------------------------------------------------------------------------
-------------------------------------------------------

directory=$1;
echo "-- PARSED DIRECTORY --"
echo $directory;
echo "-- FILES IN DIRECTORY --"
ls ../expe/$directory
cd ../..;

# Launch file preprocessing

echo "############################ FILE PREPROCESSING
############################"
th preprocess.lua -train_src data/expe/$directory/train_en.txt -train_tgt
data/expe/$directory/train_fr.txt -valid_src
data/expe/$directory/dev_en.txt -valid_tgt data/expe/$directory/dev_fr.txt
-save_data data/expe/$directory/$directory-data;

# Train machine translation engine/model

echo "############################ MT ENGINE TRAINING
############################"
```

```
th train.lua -data data/expe/$directory/$directory-data-train.t7 -
save_every_epochs 1 -save_model data/expe/$directory/$directory-model;

# Retrieve epoch with lowest perplexity and remove other epoch files

echo "############################ BEST EPOCH ############################"
cd data/utils;
best_epoch=$(python3 best_epoch.py ../expe/$directory/ 2>&1);
echo "-- BEST EPOCH -- "
echo $best_epoch;
python3 remove_tmp_epochs.py ../expe/$directory/ $best_epoch;

# Translate test subcorpus using best model

echo "############################ TRANSLATION TEST
############################"
cd ../..;
echo "Translation in progress...";
th translate.lua -model data/expe/$directory/$best_epoch -src
data/expe/$directory/test_en.txt -output data/expe/$directory/pred.txt -
replace_unk true -disable_logs true;

# Evaluate prediction (BLEU and TER scores)

echo "############################ TEST EVAL ############################"
th tools/score.lua data/expe/$directory/test_fr.txt -scorer bleu <
data/expe/$directory/pred.txt;
th tools/score.lua data/expe/$directory/test_fr.txt -scorer ter <
data/expe/$directory/pred.txt;
```

Script II : Récupération de la meilleure epoch (best_epoch.sh)

```
# ------------------------------------------------------------------------
---------------------------------------
# Utility: Parse directory containing several epochs after MT training and
return the one with lowest perplexity
# ------------------------------------------------------------------------
---------------------------------------
# Usage: python3 best_epoch.py [dir]
# ------------------------------------------------------------------------
---------------------------------------
# INPUT: directory containing several epoch files
# OUTPUT: str, name of the epoch with minimum perplexity
# ------------------------------------------------------------------------
---------------------------------------

import os, re, sys

dir = sys.argv[1]
file_list = list()

# 1. Création de la liste des fichiers avec métadonnées
for root, dirs, files in os.walk(dir):
    for file in files:
        if 'epoch' in file:
            m = re.search(r'([0-9]?[0-9])_([0-9]?[0-9]?[0-9]?[0-9]\.[0-
9][0-9])', file)
            file_list.append([file, m.group(1), m.group(2)])
```

```
# 2. Recherche de l'époque avec le minimum de perplexité
best_epoch = float(file_list[0][2])
for file in file_list:
    if float(file[2]) < best_epoch:
        best_epoch = float(file[2])
        best_file = file[0]

print(best_file)
```

Script III : Conversion des TMX en bitextes et répartition en train, dev et test (convert_and_split.py)

```
# ------------------------------------------------------------------------
------------------------------------------------------
# Utility: Prepare bilingual  TMX files for OpenNMT training
# 1/ Convert TMX files into 2 aligned raw texts (bitext)
# 2/ Split both texts into train, dev and test sub-corpora (respectively
80%, 10% and 10% of initial corpora)
# ------------------------------------------------------------------------
------------------------------------------------------
# Usage: python3 convert_and_split.py [dir name of initial TMX file]
# ------------------------------------------------------------------------
------------------------------------------------------
# Note: Languages still hard coded for now (en-fr), to be included as
parameters/retrieved on the fly
# ------------------------------------------------------------------------
------------------------------------------------------
# INPUT: directory containing bilingual (en & fr) tmx file
# OUTPUT: 2 raw texts containing all sentence in each language + 6 raw
texts: 1 dev, 1 train and 1 test in both languages
# ------------------------------------------------------------------------
------------------------------------------------------

import os, re, sys
from bs4 import BeautifulSoup

dir = sys.argv[1]

# 1. Convert TMX files into 2 aligned raw texts a.k.a. 'bitext'

def tmx2bt(dir):
    for root, dirs, files in os.walk(dir):
        for file in files:
            if "tmx" in file:
                print("converting to: "+dir+'/'+file+'_en.txt')
                print("converting to: "+dir+'/'+file+'_fr.txt')
                data_out_en = open(dir+'/'+file+'_en.txt', 'w',
encoding='utf8')
                data_out_fr = open(dir+'/'+file+'_fr.txt', 'w',
encoding='utf8')
                with open (dir+'/'+file, encoding='utf-8') as f:
                    i=0
                    for line in f:
                        if line.strip().startswith('<tuv'):
                            soup = BeautifulSoup(line, features = "lxml")
                            for a in soup.findAll('tuv', {"xml:lang":
"en"}):
                                sent_en = (a.string+'\n').encode('utf-8',
'ignore').decode('utf-8')
```

```
                                data_out_en.write(sent_en)
                            for a in soup.findAll('tuv', {"xml:lang":
"fr"}):
                                sent_fr = (a.string+'\n').encode('utf-8',
'ignore').decode('utf-8')
                                data_out_fr.write(sent_fr)
                            i+=1

# 2. Split both texts into train, dev and test sub-corpora (80%, 10% & 10%
of initial text respectively)

def split(dir):
    for root, dirs, files in os.walk(dir):
        for file in files:
            if "tmx_en.txt" in file:
                data_en = dir+'/'+file
                print("source file: "+data_en)
            if "tmx_fr.txt" in file:
                data_fr = dir+'/'+file
                print("target file: "+data_fr)

    try:
        data_en
        train_fr = open(dir+'/train_fr.txt', 'w', encoding='utf-8')
        dev_fr = open(dir+'/dev_fr.txt', 'w', encoding='utf-8')
        test_fr = open(dir+'/test_fr.txt', 'w', encoding='utf-8')
        train_en = open(dir+'/train_en.txt', 'w', encoding='utf-8')
        dev_en = open(dir+'/dev_en.txt', 'w', encoding='utf-8')
        test_en = open(dir+'/test_en.txt', 'w', encoding='utf-8')

        with open (data_en, encoding='utf-8') as f:
            ct = 0
            lines = f.readlines()
            for line in lines:
                ct += 1
                if str(ct)[-1] == '9':
                    dev_en.write(line)
                if str(ct)[-1] == '0':
                    test_en.write(line)
                else:
                    train_en.write(line)

        with open (data_fr, encoding='utf-8') as f:
            ct = 0
            lines = f.readlines()
            for line in lines:
                ct += 1
                if str(ct)[-1] == '9':
                    dev_fr.write(line)
                if str(ct)[-1] == '0':
                    test_fr.write(line)
                else:
                    train_fr.write(line)
    except:
        pass

# 0. Main

if __name__ == '__main__':
    tmx2bt(dir)
    split(dir)
```

Script IV : Calcul du score BLEU sur les différents corpus testés (bleu_calculation.py)

```
import os, glob
from bleu_utils import *
from nltk.translate.bleu_score import SmoothingFunction

# 1. Traitement des références : On enregistre les traductions de référence
sous forme de liste
path = "reference/"
path_ref_list = list()

for infile in glob.glob(os.path.join(path, '*.txt')):
    path_ref = infile.replace("\\", "/")
    result = txt_2_str_4_bleu_ref(path_ref)
    path_ref_list.append(result)

# 2. Parsing des candidats : On parcourt l'arboresence avec walk
# Structure : parcourt du dossier racine "candidate", puis de chacun des 5
sous-dossiers
# Résultat : impression du score pour chaque fichier candidat par rapport à
la traduction de référence
path2 = "candidate/"
smoothie = SmoothingFunction().method4
for root, dirnames, filenames in os.walk(path2):
    for subdir in dirnames:
        subpath = path2+subdir

        for root, dirnames, filenames in os.walk(subpath):
            i = 0
            for filename in filenames:
                file_subpath = subpath+"/"+filename
                candidate_data =
txt_2_str_4_bleu_cand(file_subpath.replace("\\", "/"))
                if "Blackboard" in file_subpath:
                    score = sentence_bleu(path_ref_list[i], candidate_data,
smoothing_function=smoothie)
                else:
                    score = sentence_bleu(path_ref_list[i], candidate_data)
                print("{} :\t{}".format(file_subpath, score))
                i += 1
```