Optimiser le choix des exemples pour la traduction automatique augmentée par des mémoires de traduction

Maxime Bouthors, Josep Crego, François Yvon

Optimiser le choix des exemples pour la traduction automatique augmentée par des mémoires de traduction

Maxime Bouthors[1, 2] Josep Crego[1] François Yvon[2]
(1) Sorbonne Université, CNRS, ISIR, F-75005 Paris, France
(2) ChapsVision, 92150, Suresnes
bouthors@isir.upmc.fr, jcrego@chapsvision.com, yvon@isir.upmc.fr

RÉSUMÉ

La traduction neuronale à partir d'exemples s'appuie sur l'exploitation d'une mémoire de traduction contenant des exemples similaires aux phrases à traduire. Ces exemples sont utilisés pour conditionner les prédictions d'un décodeur neuronal. Nous nous intéressons à l'amélioration du système qui effectue l'étape de recherche des phrases similaires, l'architecture du décodeur neuronal étant fixée et reposant ici sur un modèle explicite d'édition, le Transformeur « multi-Levenshtein ». Le problème considéré consiste à trouver un ensemble optimal d'exemples similaires, c'est-à-dire qui couvre maximalement la phrase source. En nous appuyant sur la théorie des fonctions sous-modulaires, nous explorons de nouveaux algorithmes pour optimiser cette couverture et évaluons les améliorations de performances auxquelles ils mènent pour la tâche de traduction automatique.

ABSTRACT

Optimizing example selection for retrieval-augmented machine translation with translation memories

Retrieval-augmented machine translation leverages examples from a translation memory by retrieving similar instances. These examples are used to condition the predictions of a neural decoder. We aim to improve the upstream retrieval step and consider a fixed downstream edit-based model : the multi-Levenshtein Transformer. The task consists of finding a set of examples that maximizes the overall coverage of the source sentence. To this end, we rely on the theory of submodular functions and explore new algorithms to optimize this coverage. We evaluate the resulting performance gains for the machine translation task.

MOTS-CLÉS : Traduction Automatique, Recherche d'Information, Mémoires de Traduction, Fonctions Sous-Modulaires, Traduction à partir d'Exemples.

KEYWORDS: Machine Translation, Information Retrieval, Translation Memories, Submodularity, Example-based Translation.

1 Introduction

De nombreux travaux récents s'intéressent à la génération augmentée par des exemples (Li et al., 2022). En traduction, l'utilisation d'exemples remonte aux méthodes de traduction assistée par ordinateur par des traducteurs professionnels (Bowker, 2002) : éditer des segments très similaires à la phrase de référence permet d'accélérer la traduction. Cette idée est au fondement des méthodes

basées sur des exemples (Nagao, 1984; Somers, 1999; Carl et al., 2004). Elle est adaptée aux systèmes statistiques par (Koehn & Senellart, 2010) et plus récemment aux méthodes neuronales.

Il existe en fait de nombreuses manières d'exploiter les exemples : la traduction conditionnelle qui introduit un système d'attention sur les exemples (Gu et al., 2018; Bulte & Tezcan, 2019; Hoang et al., 2022) ; l'affinage « léger » sur un ensemble d'exemples pour faire de la micro-adaptation (Farajian et al., 2017) ; les méthodes intégrant des exemples dans le contexte de grands modèles de langue (LLM) génératifs multilingues (Moslem et al. (2023), *inter alia*) ; l'édition directe du meilleur exemple similaire (Gu et al., 2019). Nous discutons ces études dans la section 2.

Ici, nous nous intéressons au transformeur « multi-Levenshtein » (Bouthors et al., 2023), un modèle d'édition qui combine $k(\geq 1)$ exemples pour calculer une traduction. Cette caractéristique le rend sensible à la qualité des exemples récupérés. En particulier, supposant fixé k le nombre de phrases à récupérer, nous cherchons à répondre à la question : comment identifier un ensemble optimal de k exemples ? Trouver exactement ce meilleur ensemble conditionnellement au modèle et à la phrase source est difficile, ce qui implique de considérer des heuristiques. Pour les construire, nous faisons l'hypothèse qu'un ensemble de phrases parallèles couvrant (en source) la phrase à traduire fournit des exemples couvrant (en cible) la traduction à produire. En dépit de ses limites, liées à des phénomènes linguistiques bien connus (variation lexicale, divergences morphologiques ou syntaxiques entre langues source et cible, etc.), cette hypothèse est adoptée dans les travaux de l'état-de-l'art.

Notre contribution principale est alors l'étude de plusieurs manières de définir la notion de couverture et de rechercher des k meilleurs exemples dans une mémoire de traduction. En tirant parti de la théorie des fonctions sous-modulaires, dont une sous-classe correspond à une notion très générique de couverture, nous analysons dans un cadre unifié les avantages comparés de ces différentes propositions, et évaluons leur impact dans une tâche de traduction multidomaines.

2 Travaux Connexes

De nombreux efforts pour intégrer des exemples dans la génération de textes ont été menés ces dernières années (Li et al., 2022). Au-delà des améliorations de performances, la possibilité de présenter aux utilisateurs améliore la transparence des décisions qui sont prises (Rudin, 2019). En traduction automatique, les exemples récupérés d'une mémoire de traduction sont fournis au modèle comme un contexte supplémentaire, par exemple en concaténant le côté cible des exemples au texte source (Bulte & Tezcan, 2019), ou en tirant parti à la fois de la source et de la cible (Pham et al., 2020). Gu et al. (2018); Xia et al. (2019); He et al. (2021b) considèrent des stratégies plus sophistiquées pour enrichir le contexte source.

Si beaucoup de travaux se limitent à considérer les $k \geq 1$ exemples les plus similaires, Cheng et al. (2022); Agrawal et al. (2023); Sia & Duh (2023) cherchent à trouver un ensemble d'exemples complémentaires entre eux. Le premier travail utilise l'algorithme de *Maximum Marginal Relevance* (MMR) (Goldstein & Carbonell, 1998), tandis que les deux autres proposent une forme de maximisation de couverture. Gupta et al. (2023) donne une formulation générale du problème de couverture, l'appliquant à l'apprentissage en contexte (*in context learning*) sur des tâches diverses.

Une autre extension de cette approche exploite des corpus monolingues, en recherchant les exemples directement dans la langue cible. Cai et al. (2021) proposent un modèle de recherche et de traduction unique entraîné de bout-en-bout dont la procédure de recherche translingue est optimisée pour

retourner des exemples utiles pour la Traduction Automatique (TA).

La plupart de ces travaux reposent sur des modèles de génération auto-régressifs (AR), impliquant que les exemples intégrés au contexte n'ont qu'un effet indirect sur la sortie. L'utilisation d'une mémoire de traduction avec un décodeur non auto-régressif (NAR) est étudiée par Niwa et al. (2022); Xu et al. (2023); Zheng et al. (2023) qui adaptent le transformeur de Levenshtein (Gu et al., 2019) pour éditer directement l'exemple le plus similaire en une traduction de la phrase source. Bouthors et al. (2023) étendent cette technique à plusieurs exemples.

Une autre approche consiste à utiliser des similarités au niveau des contextes de génération, c'est-à-dire d'états cachés du décodage des mémoires de traduction, plutôt qu'au niveau des phrases (Zhang et al., 2018). He et al. (2021a); Khandelwal et al. (2021), entre autres, utilisent des méthodes de plus proches voisins sur des contextes. La prédiction du token suivant est alors guidée par ceux trouvés dans des contextes proches. Diverses extensions sont apportées par Zheng et al. (2021); Meng et al. (2022); Martins et al. (2022).

Il est enfin difficile d'ignorer l'essor des grands modèles de langue multilingues (LLM) qui, amorcés par un contexte contenant une description de la tâche à accomplir et des exemples, peuvent générer des traductions de qualité. Cette approche a été testée sur de nombreux LLM dans le but de mettre en évidence leur capacité à traiter de multiples tâches. Pour ce qui concerne la TA, plusieurs travaux étudient l'impact du contexte d'entrée, en cherchant à optimiser le nombre d'exemples et leur sélection (Vilar et al., 2023; Zhang et al., 2023; Hendy et al., 2023; Bawden & Yvon, 2023). Voir également sur ces questions (Moslem et al., 2023; Mu et al., 2023; Agrawal et al., 2023; Sia & Duh, 2023; M et al., 2023).

3 Le modèle « multi-Levenshtein »

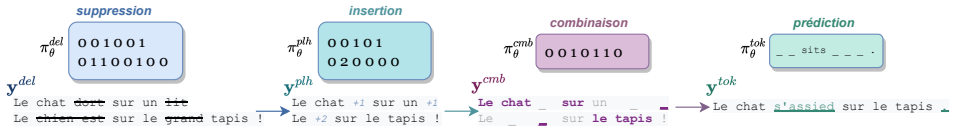

FIGURE 1 – Première étape de décodage de $\texttt{TM}^k\texttt{-LevT}$. Les deux exemples qui sont édités sont y_1 : « le chat dort sur un lit » et y_2 :« le chien est sur le grand tapis ». Les insertions prédites à l'étape 2 (insertion) sont représentées par des entiers, puis matérialisées par des '_' à l'étape 3 (combinaison).

Nous nous intéressons au modèle du transformeur multi-Levenshtein, $\texttt{TM}^k\texttt{-LevT}$, (Bouthors et al., 2023), qui repose sur un modèle explicite d'édition d'exemples. L'algorithme de décodage est représenté sur la Figure 1. Il s'agit d'un modèle d'édition type transformeur (Vaswani et al., 2017) qui prend en entrée k couples de phrases exemples cibles $\{y_1, \ldots, y_k\}$, et les édite conditionnellement à la source x en quatre étapes :

1. **Suppression** : pour chaque exemple cible y_i, délétion de tokens ;
2. **Insertion** : pour chaque exemple cible y_i, insertion de tokens vides PLH entre chaque token ;
3. **Combinaison** : combinaison des k exemples cibles en s'appuyant sur un multi-alignement qui permet de déterminer, pour chaque position, l'origine (parmi $y_1 \ldots y_k$) du token à conserver ;
4. **Prédiction** : pour chaque position comprenant un PLH, prédiction du mot à insérer.

Le calcul des associations entrées/sorties est réalisé par une architecture encodeur-décodeur non-autorégressive, dans lequel on remplace la couche de prédiction de mots habituelle par quatre couches linéaires (une pour chaque opération) qui projettent les représentations latentes sur l'ensemble des opérations possibles. Par exemple, pour l'insertion (étape 2), pour une source x et des exemples $y_1, \cdots, y_k$:

$$\texttt{insertion}^* = \arg\max \texttt{Insertion}(\texttt{Décodeur}(\texttt{Encodeur}(x), y_1, \cdots, y_k)) \quad (1)$$

Le modèle est entraîné par apprentissage par imitation (Daumé et al., 2009; Ross et al., 2011), en utilisant comme politique experte [1] la série d'opérations qui maximise la copie des tokens qui sont à la fois présents dans les exemples d'entrée et dans la référence. Autrement dit, cette politique optimale s'appuie sur une notion de couverture optimale. Pour déterminer cette politique, un algorithme d'alignement calcule la manière optimale de faire correspondre les exemples et la référence.

On se reportera à (Bouthors et al., 2023) pour une présentation détaillée de cette architecture de base et de diverses extensions (réalignement, pré-apprentissage) qui lui permettent de tirer efficacement parti de plusieurs exemples similaires. L'essentiel étant de noter que dans son principe même, cette architecture est particulièrement sensible aux exemples qui sont fournis en entrée du système.

4 Recherche d'Information dans une Mémoire de Traduction

4.1 Recherche de Phrases Similaires (RPS)

Supposons que l'on ait accès à une mémoire de traduction $\mathcal{D} = \{(x_1, y_1), \cdots (x_N, y_N)\}$ et soit x une phrase source que l'on cherche à traduire. Le cadre classique consiste à évaluer indépendamment chaque candidat selon un score de similarité $s(x_i, x)$ et récupérer les k exemples les plus similaires. s peut être un score lexical (similarité de Jaccard, TF-IDF, BM25, distance d'édition, rappel n-gramme, etc.) ou sémantique (similarité cosinus entre deux plongements). Les premiers (Jaccard, BM25) reposent sur des algorithmes simples et servent souvent à filtrer un premier ensemble T de candidats, que l'on peut ensuite évaluer avec des fonctions de comparaison plus sophistiquées.

Cependant, récupérer les k meilleurs candidats peut s'avérer sous-optimal, par exemple lorsque tous ces exemples sont très similaires entre eux. Pour évaluer globalement l'ensemble des candidats retournés par la RPS, des stratégies visant à introduire de la diversité (par exemple via l'algorithme de *Maximum Marginal Relevance*, MMR) ou des contraintes de couverture peuvent alors être déployées. Nous considérons la deuxième stratégie, la première étant documentée dans (Cheng et al., 2022).

4.2 Fonctions sous-modulaires et couverture

Par analogie aux formalisations développées dans un cadre de recherche d'information pour trouver des résultats variés (Lin & Bilmes, 2011; Krause & Golovin, 2014), nous nous appuyons sur la théorie des fonctions sous-modulaires pour formaliser la RPS basée sur la maximisation de la couverture

1. En apprentissage par imitation, la politique « experte » est celle que l'apprenti cherche à reproduire (Knyazeva et al., 2018) : elle indique ici les actions optimales qu'il faut effectuer pour éditer les exemples et générer la traduction de référence. Comme elle n'est pas observée dans les données d'apprentissage, il faut la calculer en s'appuyant sur diverses heuristiques.

de la phrase source. Nous commençons par rappeler quelques définitions, avant de présenter les algorithmes de sélection d'exemples.

4.2.1 Définitions

Définition 1 (Sous-modularité)**.** *Soient Ω un ensemble et $f : 2^\Omega \to \mathbb{R}$, f est **sous-modulaire** si $\forall X, Y$ tels que $X \subset Y \subset \Omega, \forall z \in \Omega \setminus Y$:*

$$f(X \cup \{z\}) - f(X) \geq f(Y \cup \{z\}) - f(Y)$$

Intuitivement, cette définition exprime que le rendement marginal de f est décroissant : plus l'ensemble en entrée de f est grand, plus les incréments de f induits par l'ajout de nouveaux éléments sont faibles. Une classe de fonctions sous-modulaires bien documentée est la classe des fonctions de couverture pondérée (Krause & Golovin, 2014).

Définition 2 (Couverture pondérée)**.** *Soit $N \in \mathbb{N}$ et $v^{(n)}(z)_{n\in[1,N]}$ une séquence de poids réels associée à $z \in \Omega$, correspondant à des aspects (de x) : $v^{(n)}(z)$ évalue à quel point z couvre l'aspects n. Une **fonction de couverture pondérée** associe à un sous-ensemble Z de Ω :*

$$f(Z) = \sum_{n=1}^{N} \max_{z\in Z} v^{(n)}(z)$$

Dans notre application, Ω est un ensemble d'exemples (source et références jointes) et N dénombre des aspects importants de la source x qu'il faut couvrir. Cet ensemble d'aspects peut être le sac-de-mots associé à $x = (x_1, \cdots, x_N)$, les indices de la séquence, l'ensemble des n-grammes ou des sous-arbres syntaxiques de taille bornée, etc. L'objectif est ensuite de trouver un ensemble Z de taille k qui maximise $f(Z)$, c.-à-d. qui garantit une couverture maximale pour chaque aspect.

Dans cette définition, le choix d'un opérateur $\max$ s'appliquant uniformément à tous les aspects est problématique et peut conduire à récupérer des phrases dans Z qui n'ont que peu de pertinence pour la TA, voir annexe B. En traduction, il est en effet plus important de couvrir certains aspects que d'autres (par exemple des lexèmes rares dans une représentation sac-de-mots). Pour pallier ce problème, nous introduisons une nouvelle fonction sous-modulaire.

Définition 3 (Couverture pondérée lissée)**.** *Soit $N \in \mathbb{N}$, $Z = \{z_1, \cdots, z_{|Z|}\}$ et $(v_i^{(n)})_{n\in[1,N]}$ une séquence de poids réels pour $z_i \in Z$. Une **fonction de couverture pondérée lissée** de paramètre $\lambda \in [0, 1]$ est définie par :*

$$f(Z) = \sum_{n=1}^{N} \sum_{j=1}^{|Z|} \lambda^{j-1} v^{(n)}_{g^{(n)}(j)}, \tag{2}$$

avec $g^{(n)}$ une permutation telle que $i < j \Rightarrow v^{(n)}_{g^{(n)}(i)} \geq v^{(n)}_{g^{(n)}(j)}$, ordonnant les z_i selon les $v_i^{(n)}$.

La preuve de sa sous-modularité est en Annexe C.1. Avec cette nouvelle définition, un aspects n déjà couvert continue de contribuer au calcul de f, mais avec un coefficient qui diminue exponentiellement. Pour $\lambda = 0$, f correspond à la définition (2) et, pour $\lambda = 1$, la fonction devient modulaire et sa maximisation revient à choisir indépendamment les k exemples les plus couvrants.

4.2.2 Maximisation de la couverture

Maximiser f pour $\lambda < 1$ est NP-complet (Krause & Golovin, 2014). On ne connaît pas d'algorithme exact meilleur que l'énumération exhaustive. L'algorithme glouton 1 est la façon standard de maximiser une fonction sous-modulaire. Cependant, la combinaison linéaire figurant dans la définition de f à l'équation (2) devant être recalculée pour chaque candidat restant z, cet algorithme a une complexité temporelle $O(k^2N|T| + N|T|\log|T|)$ ainsi qu'un coût spatial lié aux permutations de tri g.

Pour améliorer la complexité, nous considérons l'algorithme 2 d'Agrawal et al. (2023), replacé ici dans le cadre de la théorie des fonctions sous-modulaires afin de garantir une borne inférieure.

Algorithme 1: Maximisation gloutonne d'une fonction sous-modulaire monotone

Données: source x, fonction sous-modulaire (de couverture de pondérée) f, ensemble de candidats T, nombre d'exemples souhaité k.
Résultat: Z
$Z \leftarrow \emptyset$;
tant que $|Z| < k$ **faire**
  $z^* \leftarrow \arg\max_{z \in T \setminus Z} f(Z \cup \{z\})$;
  $Z \leftarrow Z \cup \{z^*\}$;
fin
retourne Z ;

Algorithme 2: Maximisation gloutonne d'une fonction de couverture pondérée lissée

Données: source x, fonction sous-modulaire de couverture de x pondérée lissée f de facteur de sous-pondération λ, poids de couverture $v^{(n)}(z)$ pour $z \in T$ ensemble de candidats, nombre d'exemples souhaité k.
Résultat: Z
$Z \leftarrow \emptyset$;
$W \leftarrow (1, \ldots, 1)$; /* $|W| = N$ */
tant que $|Z| < k$ **faire**
  $z^* \leftarrow \arg\max_{z \in T \setminus Z} W^T v(z)$; /* séléctionner z^* */
  $Z \leftarrow Z \cup \{z^*\}$;
  $I^* \leftarrow \{n : v^{(n)}(z^*) > 0\}$; /* aspects couverts par z^* */
  pour $n \in I^*$ **faire**
    $W_n \leftarrow \lambda W_n$; /* sous-pondération des éléments couverts */
  fin
  si $\lambda = 0$ ***et*** $W = (0, \ldots, 0)$ **alors**
    $W \leftarrow (1, \ldots, 1)$; /* réinitialiser W */
  fin
fin
retourne Z ;

L'algorithme 2 a une complexité temporelle $O(kN|T|)$. Il se rapproche de l'algorithme 1, à la différence près que les $v_i^{(n)}$ ne sont pas triés. Nous montrons en annexe C.2 que l'écart entre la solution retournée par l'algorithme 2 et la solution optimale est borné.

5 Cadre Expérimental

5.1 Données

Nous considérons des données anglais-français sur un panel de 6 corpus de domaine varié : ECB, EMEA, Europarl, JRC-Acquis, Ubuntu, Wikipedia [2]. Cela correspond à un sous-ensemble des données utilisées par Xu et al. (2023) dont nous reprenons la même partition entraînement/test. Une caractéristique des données de test est qu'elles ont été partitionnées en deux ensembles de 1000 phrases pour chaque domaine. Le premier ensemble (*test-0.4*) contient des phrases sources pour lesquelles le plus proche voisin dans la TM (au sens de la similarité de Levenshtein [3]) est à une distance entre 0,4 et 0,6. Pour le second (*test-0.6*), le plus proche voisin a un score d'au moins 0,6. Cela permet de différencier les comportements entre les échantillons avec des *« bonnes »* correspondances, et ceux avec des correspondances *« médiocres »*. Le tableau 1 résume les principales statistiques de ces corpus.

domaine	ECB	EME	Epp	JRC	Ubu	Wiki	**tout**
taille	195k	373k	2,0M	503k	9k	803k	3,9M
longueur moyenne	29,2	16,7	26,6	28,8	5,2	19,6	21,0

TABLE 1 – Nombre d'échantillons et longueur moyenne des phrases d'entraînement.

5.2 Scores de Couverture

Dans un premier temps, nous utilisons notre propre implémentation de BM25 (Robertson & Jones, 1976) pour récupérer les $T = 100$ meilleurs candidats. Ensuite, nous considérons plusieurs fonctions de couverture avec des scores et pondérations différents.

Couverture sac-de-mot (SDM) : Les aspects sont les termes sac-de-mot t_n de la source x et le poids $v^{(n)}(z)$ correspond au minimum du nombre d'occurrences de t_n dans le candidat z et dans x. Cette notion correspond au rappel modifié [4] : les termes ne peuvent pas être couverts plus que leur nombre d'occurrences dans la source.

Couverture 4-gramme ou moins (NGM) : Les aspects sont les 1-4-grammes t_n de x, et $v^{(n)}(z)$ est le minimum du nombre d'occurrences dans x et dans z.

Couverture par distance de Levenshtein (DL) : Les aspects sont les indices de x, et $v^{(n)}(z)$ vaut 1 (0 sinon) si et seulement si x_n appartient à une sous-chaîne copiée en calculant la distance de Levenshtein entre x et z. Puisqu'il peut exister plusieurs alignements optimaux, on peut soit calculer l'ensemble des sous-chaînes optimales et marginaliser pour chaque indice, soit échantillonner une solution et l'utiliser pour construire les $v^{(n)}(z)$.

Les $v^{(n)}(z)$ sont normalisés de deux manières différentes :

Normalisation par cardinalité : Pour SDM et NGM $v^{(n)}(z)$ est divisé par N le nombre d'aspects. Pour DL, on choisit de normaliser par le maximum de la taille de x et z afin de retrouver la formule

2. Les données sont en libre accès sur le site opus https://opus.nlpl.eu
3. Définie pour deux chaînes x, y par $1 - \frac{d(x,y)}{\max |x|,|y|}$, avec d la distance de Levenshtein.
4. "Modified recall", par analogie à la précision modifiée du score BLEU.

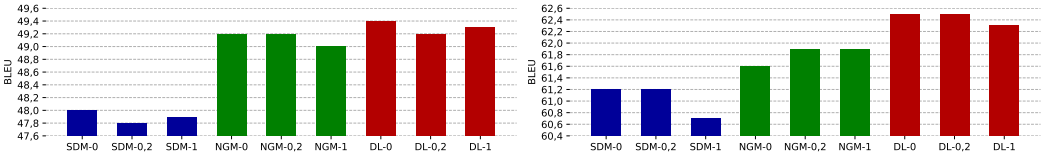

FIGURE 2 – Scores BLEU moyens selon le score de la similarité DL, pour $\lambda \in \{0; 0{,}2; 1\}$ sur *test-0.4* (gauche) et *test-0.6* (droite).

classique quand $\lambda = 1$.

Normalisation par rareté (IDF) : Chaque valeur $v^{(n)}(z)$ est normalisée avec des pondérations IDF, c.-à-d. que, pour SDM et DL, chaque mot w de x reçoit un poids $\mathrm{IDF}(w) / \sum_{w' \in x} \mathrm{IDF}(w)$. Pour NGM, l'indexation des n-grammes jusqu'à l'ordre 4 étant coûteuse, nous avons par simplification remplacé l'IDF de chaque n-gramme par la moyenne des IDF des termes qu'il contient.

Enfin, dans le cadre de l'introduction des fonctions de *couverture pondérée lissées*, nous étudions différentes valeurs de $\lambda \in \{0; 0{,}2; 0{,}5; 1\}$.

Recherche constrastive Nous comparons aussi nos résultats à MMR (Cheng et al., 2022) avec $\alpha = 0{,}3$ qui fait un compromis entre pertinence et diversité des exemples.

5.3 Métriques

Nous calculons les scores BLEU (Papineni et al., 2002) avec SacreBLEU (Post, 2018) [5]. Nous étudions également trois métriques calculées en comparant les phrases cibles récupérées à la référence : la couverture, la pertinence et la longueur des phrases. La couverture est calculée selon un rappel modifié, comme pour la fonction sous-modulaire unigramme avec $\lambda = 0$ sur les phrases tokénisées [6]. La pertinence est la précision sac-de-mot, c.-à-d. la proportion de termes utiles dans les exemples rapportée à la somme des longueurs des exemples. La longueur est calculée sur les phrases tokenisées.

6 Résultats

Rôle du score choisi : Nous effectuons une recherche pour les scores SDM, NGM et DL avec normalisation cardinale et $\lambda \in \{0; 0{,}2; 1\}$. Les histogrammes des scores BLEU de la figure 2 montrent : (1) La supériorité de la distance de Levenshtein (DL) sur les deux autres scores, même si NGM reste compétitif sur certains domaines (voir tableau 4 dans l'annexe D pour les résultats par domaine) ; (2) À l'exception de NGM-0 (*test-0.6*) maximiser la couverture avec $\lambda = 0$ est en moyenne préférable et conduit à un gain de +0,1 (resp. +0,2) pour *test-0.4* (resp. *test-0.6*) par rapport à $\lambda = 1$.

5. signature : `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.1.0`;
6. Nous utilisons les scripts de Moses (https://github.com/moses-smt/)

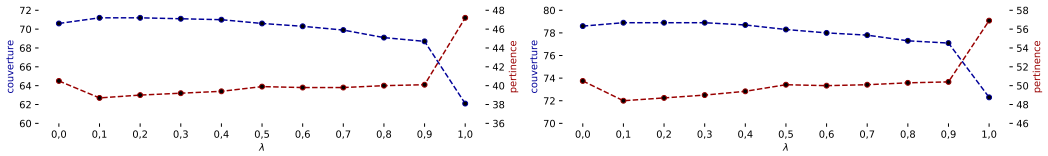

FIGURE 3 – Couverture et pertinence moyenne selon la valeur de λ pour DL sur *test-0.4* (gauche) et *test-0.6* (droite).

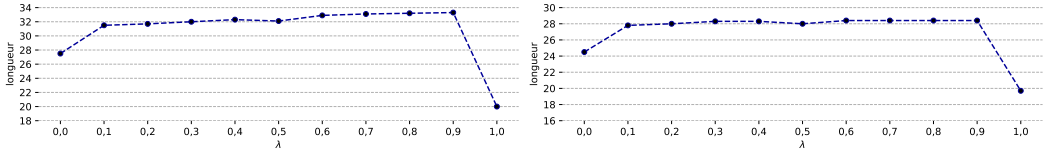

FIGURE 4 – Longueur moyenne selon la valeur de λ pour DL sur *test-0.4* (gauche) et *test-0.6* (droite).

Rôle de λ : Nous regardons l'impact de λ pour DL (Distance de Levenshtein) figures 3 et 4 en termes de couverture, pertinence et longueur. Nous trouvons un compromis entre pertinence et couverture. L'intervalle $[0{,}1; 0{,}9]$ est très stable. Nous observons une singularité à $\lambda = 0$. La couverture est légèrement plus faible que $\lambda = 0{,}1$. Le cas $\lambda = 1$ est singulier car ayant la couverture (resp. pertinence) la plus faible (resp. la plus élevée), ainsi que la plus faible longueur moyenne. Quant au score **BLEU** (voir figure 5), on observe un comportement différent entre les correspondances *médiocres* (*test-0.4*) et *bonnes* (*test-0.6*). $\lambda = 0$ semble en général mieux sur *test-0.4*, même si aucune tendance ne s'en dégage. Sur *test-0.6*, on observe une courbe en cloche avec une inflexion à $\lambda = 0$. La tendance semble indiquer que $\lambda = 0{,}5$ produit les meilleurs résultats, et cela même sur *test-0.4*. À noter qu'il y a des différences entre domaines (voir Annexe D).

Comparaison avec MMR Nous comparons les méthodes DL pour $\lambda \in \{0; 0{,}5; 1\}$ avec la méthode contrastive MMR. Sur la figure 6, nous observons de très légères différences entre les méthodes de recherche. MMR est légèrement mieux (+0,1 BLEU) sur *test-0.4*, mais DL-0,5 surpasse MMR de 0,1 sur *test-0.6*. Aucune des deux méthodes ne semble particulièrement supérieure.

Rôle de la normalisation : Par défaut, la normalisation se fait sur le nombre d'aspects à couvrir, sauf pour DL où il s'agit du maximum entre la longueur de la source et de l'exemple. Lorsqu'on

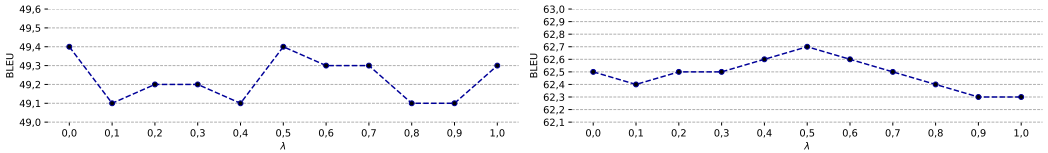

FIGURE 5 – Score BLEU moyen en fonction de λ pour DL sur *test-0.4* (gauche) et *test-0.6* (droite).

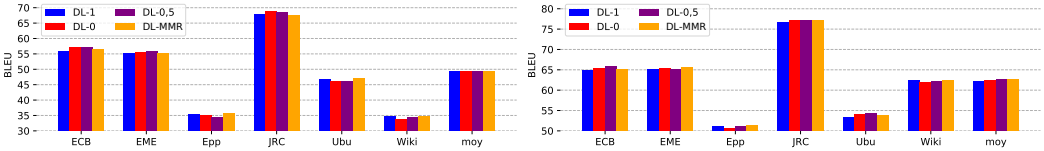

FIGURE 6 – Score BLEU avec $\lambda \in \{0; 0{,}5; 1\}$ et MMR sur *test-0.4* (gauche) et *test-0.6* (droite).

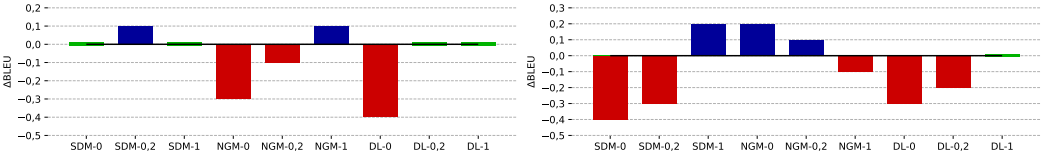

FIGURE 7 – Différence moyenne de BLEU entre normalisation IDF et cardinale, avec $\lambda \in \{0; 0{,}2; 1\}$ et SDM, NGM, DL sur *test-0.4* (gauche) et *test-0.6* (droite).

introduit une normalisation IDF, on observe un effet en moyenne négatif sur le score BLEU (voir figure 7). On peut expliquer ce comportement par le fait que le transformeur Multi-Levenshtein apprend à copier un maximum de tokens, même si ceux-ci sont fréquents. Autrement dit, il favorise la quantité à la qualité. Il vaut donc probablement mieux fournir des exemples plus couvrants, même s'il s'agit de termes communs.

7 Conclusion

Dans cet article, nous étudions comment optimiser la sélection d'un ensemble varié d'exemples dans une mémoire de traduction. Par analogie des travaux en recherche d'information, nous nous appuyons sur la théorie des fonctions sous-modulaires. Dans certaines configurations nous observons un léger gain avec l'utilisation de ce paradigme, avec de grandes variations selon le domaine. Une difficulté de cette approche est qu'elle sélectionne des exemples plus longs que la méthode de base, ce qui rend plus difficile leur édition conjointe et limite les améliorations des scores de traduction.

Dans le futur, nous souhaitons continuer à étudier la relation entre choix des exemples et modèle d'édition, par exemple en entraînant le modèle de recherche d'information conjointement avec le modèle de traduction. Une autre piste consiste à chercher à imposer une contrainte de longueur aux phrases couvrantes, afin qu'elles soient plus exploitables. Enfin, $\texttt{TM}^k\texttt{-LevT}$ apprend à maximiser la couverture de la cible sans se soucier de la rareté des mots couverts. Peut-être vaut-il mieux privilégier l'utilité des mots couverts plutôt que leur nombre, en incitant le modèle à conserver des termes qu'il aurait du mal à regénérer.

8 Remerciements

Ce projet a été partiellement financé par l'ANR dans le cadre du projet TraLaLam (ANR-23-IAS1-0006). Il a également bénéficié des resources HPC/AI de GENCI-IDRIS (2022-AD011013583 et 2023-AD010614012).

Références

AGRAWAL S., ZHOU C., LEWIS M., ZETTLEMOYER L. & GHAZVININEJAD M. (2023). In-context examples selection for machine translation. In A. ROGERS, J. BOYD-GRABER & N. OKAZAKI, Éds., Findings of the Association for Computational Linguistics : ACL 2023, p. 8857–8873, Toronto, Canada : Association for Computational Linguistics. DOI : 10.18653/v1/2023.findings-acl.564.

BAWDEN R. & YVON F. (2023). Investigating the translation performance of a large multilingual language model : the case of BLOOM. In M. NURMINEN, J. BRENNER, M. KOPONEN, S. LATOMAA, M. MIKHAILOV, F. SCHIERL, T. RANASINGHE, E. VANMASSENHOVE, S. A. VIDAL, N. ARANBERRI, M. NUNZIATINI, C. P. ESCARTÍN, M. FORCADA, M. POPOVIC, C. SCARTON & H. MONIZ, Éds., Proceedings of the 24th Annual Conference of the European Association for Machine Translation, p. 157–170, Tampere, Finland : European Association for Machine Translation.

BOUTHORS M., CREGO J. & YVON F. (2023). Towards example-based NMT with multi-Levenshtein transformers. In H. BOUAMOR, J. PINO & K. BALI, Éds., Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, p. 1830–1846, Singapore : Association for Computational Linguistics. DOI : 10.18653/v1/2023.emnlp-main.113.

BOWKER L. (2002). Computer-aided translation technology : A practical introduction. University of Ottawa Press.

BULTE B. & TEZCAN A. (2019). Neural fuzzy repair : Integrating fuzzy matches into neural machine translation. In A. KORHONEN, D. TRAUM & L. MÀRQUEZ, Éds., Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, p. 1800–1809, Florence, Italy : Association for Computational Linguistics. DOI : 10.18653/v1/P19-1175.

CAI D., WANG Y., LI H., LAM W. & LIU L. (2021). Neural machine translation with monolingual translation memory. In C. ZONG, F. XIA, W. LI & R. NAVIGLI, Éds., Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers), p. 7307–7318, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-long.567.

CARL M., WAY A. & DAELEMANS W. (2004). Recent advances in example-based machine translation. Computational Linguistics, **30**, 516–520. DOI : 10.1162/0891201042544866.

CHENG X., GAO S., LIU L., ZHAO D. & YAN R. (2022). Neural machine translation with contrastive translation memories. In Y. GOLDBERG, Z. KOZAREVA & Y. ZHANG, Éds., Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, p. 3591–3601, Abu Dhabi, United Arab Emirates : Association for Computational Linguistics. DOI : 10.18653/v1/2022.emnlp-main.235.

DAUMÉ H., LANGFORD J. & MARCU D. (2009). Search-based structured prediction. Machine Learning, **75**(3), 297–325. DOI : 10.1007/s10994-009-5106-x.

FARAJIAN M. A., TURCHI M., NEGRI M. & FEDERICO M. (2017). Multi-domain neural machine translation through unsupervised adaptation. In O. BOJAR, C. BUCK, R. CHATTERJEE, C.

FEDERMANN, Y. GRAHAM, B. HADDOW, M. HUCK, A. J. YEPES, P. KOEHN & J. KREUTZER, Éds., Proceedings of the Second Conference on Machine Translation, p. 127–137, Copenhagen, Denmark : Association for Computational Linguistics. DOI : 10.18653/v1/W17-4713.

GOLDSTEIN J. & CARBONELL J. (1998). Summarization : (1) using MMR for diversity- based reranking and (2) evaluating summaries. In TIPSTER TEXT PROGRAM PHASE III : Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998, p. 181–195, Baltimore, Maryland, USA : Association for Computational Linguistics. DOI : 10.3115/1119089.1119120.

GU J., WANG C. & ZHAO J. (2019). Levenshtein transformer. In H. WALLACH, H. LAROCHELLE, A. BEYGELZIMER, F. D'ALCHÉ-BUC, E. FOX & R. GARNETT, Éds., Advances in Neural Information Processing Systems, volume 32 : Curran Associates, Inc.

GU J., WANG Y., CHO K. & LI V. O. (2018). Search Engine Guided Neural Machine Translation. Proceedings of the AAAI Conference on Artificial Intelligence, **32**(1). DOI : 10.1609/aaai.v32i1.12013.

GUPTA S., GARDNER M. & SINGH S. (2023). Coverage-based example selection for in-context learning. In H. BOUAMOR, J. PINO & K. BALI, Éds., Findings of the Association for Computational Linguistics : EMNLP 2023, p. 13924–13950, Singapore : Association for Computational Linguistics. DOI : 10.18653/v1/2023.findings-emnlp.930.

HE J., NEUBIG G. & BERG-KIRKPATRICK T. (2021a). Efficient nearest neighbor language models. In M.-F. MOENS, X. HUANG, L. SPECIA & S. W.-T. YIH, Éds., Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, p. 5703–5714, Online and Punta Cana, Dominican Republic : Association for Computational Linguistics. DOI : 10.18653/v1/2021.emnlp-main.461.

HE Q., HUANG G., CUI Q., LI L. & LIU L. (2021b). Fast and accurate neural machine translation with translation memory. In C. ZONG, F. XIA, W. LI & R. NAVIGLI, Éds., Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers), p. 3170–3180, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-long.246.

HENDY A., ABDELREHIM M., SHARAF A., RAUNAK V., GABR M., MATSUSHITA H., KIM Y. J., AFIFY M. & AWADALLA H. H. (2023). How good are GPT models at machine translation ? a comprehensive evaluation. CoRR, **abs/2302.09210**. DOI : 10.48550/ARXIV.2302.09210.

HOANG C., SACHAN D., MATHUR P., THOMPSON B. & FEDERICO M. (2022). Improving Retrieval Augmented Neural Machine Translation by Controlling Source and Fuzzy-Match Interactions. arXiv :2210.05047 [cs], DOI : 10.48550/arXiv.2210.05047.

KHANDELWAL U., FAN A., JURAFSKY D., ZETTLEMOYER L. & LEWIS M. (2021). Nearest Neighbor Machine Translation. In Proceedings of the International Conference on Learning Representations.

KNYAZEVA E., WISNIEWSKI G. & YVON F. (2018). Les méthodes « apprendre à chercher » en traitement automatique des langues : un état de l'art [a survey of learning-to-search techniques in natural language processing]. Traitement Automatique des Langues, **59**(1), 39–63.

KOEHN P. & SENELLART J. (2010). Convergence of translation memory and statistical machine translation. In V. ZHECHEV, Éd., Proceedings of the Second Joint EM+/CNGL Workshop : Bringing MT to the User : Research on Integrating MT in the Translation Industry, p. 21–32, Denver, Colorado, USA : Association for Machine Translation in the Americas.

KRAUSE A. & GOLOVIN D. (2014). Submodular function maximization. In L. BORDEAUX, Y. HAMADI & P. KOHLI, Éds., Tractability : Practical Approaches to Hard Problems, p. 71–104. Cambridge University Press.

LI H., SU Y., CAI D., WANG Y. & LIU L. (2022). A survey on retrieval-augmented text generation. CoRR, **abs/2202.01110**.

LIN H. & BILMES J. (2011). A class of submodular functions for document summarization. In D. LIN, Y. MATSUMOTO & R. MIHALCEA, Éds., Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies, p. 510–520, Portland, Oregon, USA : Association for Computational Linguistics.

M A., PUDUPPULLY R., DABRE R. & KUNCHUKUTTAN A. (2023). CTQScorer : Combining multiple features for in-context example selection for machine translation. In H. BOUAMOR, J. PINO & K. BALI, Éds., Findings of the Association for Computational Linguistics : EMNLP 2023, p. 7736–7752, Singapore : Association for Computational Linguistics. DOI : 10.18653/v1/2023.findings-emnlp.519.

MARTINS P. H., MARINHO Z. & MARTINS A. F. T. (2022). Chunk-based nearest neighbor machine translation. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, p. 4228–4245, Abu Dhabi, United Arab Emirates : Association for Computational Linguistics. DOI : https ://aclanthology.org/emnlp-22/2022.emnlp-main.284.

MENG Y., LI X., ZHENG X., WU F., SUN X., ZHANG T. & LI J. (2022). Fast nearest neighbor machine translation. In S. MURESAN, P. NAKOV & A. VILLAVICENCIO, Éds., Findings of the Association for Computational Linguistics : ACL 2022, p. 555–565, Dublin, Ireland : Association for Computational Linguistics. DOI : 10.18653/v1/2022.findings-acl.47.

MOSLEM Y., HAQUE R., KELLEHER J. D. & WAY A. (2023). Adaptive machine translation with large language models. In M. NURMINEN, J. BRENNER, M. KOPONEN, S. LATOMAA, M. MIKHAILOV, F. SCHIERL, T. RANASINGHE, E. VANMASSENHOVE, S. A. VIDAL, N. ARANBERRI, M. NUNZIATINI, C. P. ESCARTÍN, M. FORCADA, M. POPOVIC, C. SCARTON & H. MONIZ, Éds., Proceedings of the 24th Annual Conference of the European Association for Machine Translation, p. 227–237, Tampere, Finland : European Association for Machine Translation.

MU Y., REHEMAN A., CAO Z., FAN Y., LI B., LI Y., XIAO T., ZHANG C. & ZHU J. (2023). Augmenting large language model translators via translation memories. In A. ROGERS, J. BOYD-GRABER & N. OKAZAKI, Éds., Findings of the Association for Computational Linguistics : ACL 2023, p. 10287–10299, Toronto, Canada : Association for Computational Linguistics. DOI : 10.18653/v1/2023.findings-acl.653.

NAGAO M. (1984). A framework of a mechanical translation between Japanese and English by analogy principle. In A. ELITHORN & R. BANERJI, Éds., Artificial and human intelligence : Elsevier Science Publishers. B.V.

NIWA A., TAKASE S. & OKAZAKI N. (2022). Nearest neighbor non-autoregressive text generation. CoRR, **abs/2208.12496**. DOI : 10.48550/ARXIV.2208.12496.

PAPINENI K., ROUKOS S., WARD T. & ZHU W.-J. (2002). Bleu : a method for automatic evaluation of machine translation. In P. ISABELLE, E. CHARNIAK & D. LIN, Éds., Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, p. 311–318, Philadelphia, Pennsylvania, USA : Association for Computational Linguistics. DOI : 10.3115/1073083.1073135.

PHAM M. Q., XU J., CREGO J., YVON F. & SENELLART J. (2020). Priming neural machine translation. In L. BARRAULT, O. BOJAR, F. BOUGARES, R. CHATTERJEE, M. R. COSTA-JUSSÀ, C. FEDERMANN, M. FISHEL, A. FRASER, Y. GRAHAM, P. GUZMAN, B. HADDOW, M. HUCK, A. J. YEPES, P. KOEHN, A. MARTINS, M. MORISHITA, C. MONZ, M. NAGATA, T. NAKAZAWA & M. NEGRI, Éds., Proceedings of the Fifth Conference on Machine Translation, p. 516–527, Online : Association for Computational Linguistics.

POST M. (2018). A call for clarity in reporting BLEU scores. In O. BOJAR, R. CHATTERJEE, C. FEDERMANN, M. FISHEL, Y. GRAHAM, B. HADDOW, M. HUCK, A. J. YEPES, P. KOEHN, C. MONZ, M. NEGRI, A. NÉVÉOL, M. NEVES, M. POST, L. SPECIA, M. TURCHI & K. VERSPOOR, Éds., Proceedings of the Third Conference on Machine Translation : Research Papers, p. 186–191, Brussels, Belgium : Association for Computational Linguistics. DOI : 10.18653/v1/W18-6319.

ROBERTSON S. E. & JONES K. S. (1976). Relevance weighting of search terms. Journal of the American Society for Information Science, **27**(3), 129–146. DOI : https ://doi.org/10.1002/asi.4630270302.

ROSS S., GORDON G. & BAGNELL D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In G. GORDON, D. DUNSON & M. DUDÍK, Éds., Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, volume 15 de Proceedings of Machine Learning Research, p. 627–635, Fort Lauderdale, FL, USA : PMLR.

RUDIN C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence, **1**(5), 206–215. DOI : 10.1038/s42256-019-0048-x.

SIA S. & DUH K. (2023). In-context learning as maintaining coherency : A study of on-the-fly machine translation using large language models.

SOMERS H. (1999). Review article : Example-based machine translation. Machine Translation, **14**(2), 113–157. DOI : 10.1023/A :1008109312730.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. & POLOSUKHIN I. (2017). Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, p. 6000–6010, Red Hook, NY, USA : Curran Associates Inc.

VILAR D., FREITAG M., CHERRY C., LUO J., RATNAKAR V. & FOSTER G. (2023). Prompting PaLM for translation : Assessing strategies and performance. In A. ROGERS, J. BOYD-GRABER & N. OKAZAKI, Éds., Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers), p. 15406–15427, Toronto, Canada : Association for Computational Linguistics. DOI : 10.18653/v1/2023.acl-long.859.

XIA M., HUANG G., LIU L. & SHI S. (2019). Graph based translation memory for neural machine translation. Proceedings of the AAAI Conference on Artificial Intelligence, **33**(01), 7297–7304. DOI : 10.1609/aaai.v33i01.33017297.

XU J., CREGO J. & YVON F. (2023). Integrating translation memories into non-autoregressive machine translation. In A. VLACHOS & I. AUGENSTEIN, Éds., Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, p. 1326–1338, Dubrovnik, Croatia : Association for Computational Linguistics. DOI : 10.18653/v1/2023.eacl-main.96.

ZHANG B., HADDOW B. & BIRCH A. (2023). Prompting large language model for machine translation : A case study. In Proceedings of the 40th International Conference on Machine Learning, ICML'23 : JMLR.org.

ZHANG J., UTIYAMA M., SUMITA E., NEUBIG G. & NAKAMURA S. (2018). Guiding neural machine translation with retrieved translation pieces. In M. WALKER, H. JI & A. STENT, Éds., Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers), p. 1325–1335, New Orleans, Louisiana : Association for Computational Linguistics. DOI : 10.18653/v1/N18-1120.

ZHENG K., WANG L., WANG Z., CHEN B., ZHANG M. & TU Z. (2023). Towards a unified training for Levenshtein transformer. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), p. 1–5. DOI : 10.1109/ICASSP49357.2023.10094646.

ZHENG X., ZHANG Z., GUO J., HUANG S., CHEN B., LUO W. & CHEN J. (2021). Adaptive nearest neighbor machine translation. In C. ZONG, F. XIA, W. LI & R. NAVIGLI, Éds., Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers), p. 368–374, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-short.47.

A Configuration du Modèle

Nous utilisons le transformeur Multi-Levenshtein [7]. La dimension de plongement est 512 ; la taille de la couche linéaire est de 2048 ; le nombre de têtes est 8 ; le nombre de couches de l'encodeur et de décodeur est 6 ; les plongements sont tous partagés ; le taux de dropout est 0,3.

L'entraînement est fait avec un optimisateur Adam $(\beta_1, \beta_2) = (0{,}9; 0{,}98)$; un planificateur racine carrée inversée ; un taux d'apprentissage de $5e^{-4}$; un lissage d'étiquette de 0,1 ; une mise à jour à 10 000 ; une précision flottante de 16. Le nombre d'itérations est fixé à 60k. La taille de lot et le nombre de GPU sont choisis pour avoir en moyenne 450 échantillons par itération. Le modèle est pré-entraîné sur des données synthétiques décrites par Bouthors et al. (2023). Nous utilisons un vocabulaire joint de taille 32k. Nous utilisons le réalignement et le réaffinage itératif avec une pénalité de 3 sur le fait d'insérer 0 à l'étape d'insertion, et un nombre maximum d'itérations de 10. (Gu et al., 2019).

Les données d'entraînement sont les 11 domaines en-fr de Bouthors et al. (2023), dont les 6 domaines que nous avons choisis. Les exemples sont construits avec comme dans la configuration DL, c-à-d avec un préfiltrage des 100 exemples avec les meilleurs scores BM25, puis les 3 meilleurs exemples de similarité DL. Cependant, les phrases avec un score $< 0{,}4$ sont retirées comme dans le papier original.

B Illustration

Le tableau 2 illustre le compromis entre **couverture** et **pertinence** (liée au score DL individuel). Ici, nous effectuons une RPS sur un micro corpus de 11 phrases plus ou moins similaires à la source. Le terme "vert" apparaissant uniquement dans une phrase très peu pertinente (dans le sens où peu de termes de la phrase sont dans la source), un λ trop faible la récupérera pour compléter le dernier terme couvrable. Au contraire, un λ trop élevé a tendance à récupérer des phrases similaires les unes aux autres, avec une haute pertinence, mais une faible couverture.

7. disponible à https://github.com/Maxwell1447/fairseq.

méthode	couv	score DL	**source :** Le chat est assis sur le tapis vert du salon .
DL-0 à DL-0,3	0,91	0,64	Le chat est assis sur le sol .
		0,27	J' ai acheté un nouveau tapis pour le salon .
		0,08	Après une longue journée de marche , au crépuscule , je décide enfin de me reposer à côté d' un grand rocher tout vert de mousse dans la forêt à côté du domaine de Courbetin .
DL-0,4 à DL-0,6	0,82	0,64	Le chat est assis sur le sol .
		0,27	J' ai acheté un nouveau tapis pour le salon .
		0,27	Regarde ce chat , il est assis sur le comptoir .
DL-0,7 à DL-0,9	0,64	0,64	Le chat est assis sur le sol .
		0,27	Regarde ce chat , il est assis sur le comptoir .
		0,45	Le chat est assis à l' entrée .
DL-1	0,64	0,64	Le chat est assis sur le sol .
		0,45	Le chat est assis à l' entrée .
		0,36	Le chat est dans une boîte en carton .
DL-MMR	0,82	0,64	Le chat est assis sur le sol .
		0,45	Le chat est assis à l' entrée .
		0,27	J' ai acheté un nouveau tapis pour le salon .

TABLE 2 – Illustration des effets des paramètres de RPS (λ et MMR) pour le score DL, avec un compromis entre couverture globale de la source (couv) et la proximité individuelle des phrases exemples. Les termes couvrants sont soulignés.

C Démonstrations

C.1 Sous-modularité de la *couverture pondérée lissée*

Lemme 1. *Soit $f : 2^{\Omega} \to \mathbb{R}$. La propriété suivante découle de la définition de la sous-modularité de f : f sous-modulaire si et seulement si $\forall X \subseteq \Omega$, $\forall x_1, x_2 \in \Omega \setminus X$ s.t. $x_1 \neq x_2$:*

$$f(X \cup \{x_1\}) + f(X \cup \{x_2\}) \geq f(X \cup \{x_1, x_2\}) + f(X)$$

Nous utilisons le lemme 1 pour montrer que la fonction de *couverture pondérée lissée* (2) est sous-modulaire.

Démonstration. Soit $f_n(Z)$ le terme dans l'équation (2) tel que $f(Z) = \sum_{n=1}^{N} f_n(Z)$:

$$f_n(Z) = v_0 + \lambda v_1 + \cdots + \lambda^{K-1} v_{K-1}, \tag{3}$$

où les v_i sont triés dans l'ordre décroissants, et $K = |Z|$. Soient v et v' la composante en n de deux nouveaux éléments de $\Omega \setminus Z$. On considère que $v \geq v'$. On nomme i et j les indices tels que : $v_i \geq v \geq v_{i+1}$ et $v_j \geq v' \geq v_{j+1}$.

- $f_n(Z \cup \{v\}) = v_0 + \cdots + \lambda^i v_i + \lambda^{i+1} v + \lambda^{i+2} v_{i+1} + \cdots + \lambda^K v_{K-1}$
- $f_n(Z \cup \{v'\}) = v_0 + \cdots + \lambda^j v_j + \lambda^{j+1} v' + \lambda^{j+2} v_{j+1} + \cdots + \lambda^K v_{K-1}$
- $f_n(Z \cup \{v, v'\}) = v_0 + \cdots + \lambda^i v_i + \lambda^{i+1} v + \lambda^{i+2} v_{i+1} + \cdots$
  $+ \lambda^{j+1} v_j + \lambda^{j+2} v' + \lambda^{j+3} v_{j+1} + \cdots + \lambda^{K+1} v_{K-1}$

On simplifie en nommant $L = \sum_{l \leq i} \lambda^l v_l$, $M = \sum_{i < l \leq j} \lambda^l v_l$ et $R = \sum_{j < l} \lambda^l v_l$, ce qui donne :

- $f_n(Z \cup \{v\}) = L + \lambda^i v + \lambda(M + R)$
- $f_n(Z \cup \{v'\}) = L + M + \lambda^j v + \lambda R$
- $f_n(Z \cup \{v, v'\}) = L + \lambda^i v + \lambda M + \lambda^{j+1} v' + \lambda^2 R$

Pour montrer que f_n est sous-modulaire, il suffit de montrer que ce terme est positif :

$$f_n(Z \cup \{v\}) + f_n(Z \cup \{v'\}) - f_n(Z \cup \{v, v'\}) - f_n(Z) = (1 - \lambda) \left[\lambda^j v' - (1 - \lambda) R\right]$$

Puisque $\lambda \in [0, 1]$, alors $1 - \lambda > 0$. D'autre part :

$$\begin{aligned} R \leq \lambda^{j+1} v_{j+1} &\Rightarrow -(1 - \lambda) R \geq -(1 - \lambda) \lambda^{j+1} v_{j+1} \\ &\Rightarrow \lambda^j v' - (1 - \lambda) R \geq \lambda^j v' - (1 - \lambda) \lambda^{j+1} v_{j+1} \end{aligned}$$

$\lambda^j > \lambda^{j+1}(1 - \lambda) \geq 0$ et $v' \geq v_{j+1} \geq 0$, donc $\lambda^j v' \geq (1 - \lambda) \lambda^{j+1} v_{j+1}$.

$f_n(Z \cup \{v\}) + f_n(Z \cup \{v'\}) - f_n(Z \cup \{v, v'\}) - f_n(Z) \geq 0$ donc f_n sous-modulaire. Puisque f est une somme de fonction sous-modulaire, alors f sous-modulaire. □

C.2 Majoration de l'erreur d'approximation

Théorème 1. *Soit f une fonction de couverture pondérée lissée de paramètre λ. Soit Z la solution de l'algorithme 2 et k le nombre d'exemples :*

$$f(Z) \geq \frac{1}{k} \left(\sum_{j=0}^{k-1} \lambda^j \right) \max_{\bar{Z} : |\bar{Z}| = k} f(Z) \tag{4}$$

Si $\lambda = 1$, la solution est optimale. Si $\lambda = 0$, elle est au pire $\frac{1}{k}$ de l'optimum. Dans le cas où les poids de couverture sont booléens (0 ou 1), et $\lambda = 0$, l'algorithme 2 est équivalent au 1 (hormis la partie de réinitialisation), ce qui nous permet de retomber sur la borne bien connue de $(1 - 1/e)$. Nous montrons également que dans des cas limites spécifiques, cette borne peut être atteinte.

Démonstration. L'équation (4) se démontre via les Lemmes (2) et (4). Nous adoptons la notation suivante : $\Delta(v|Z) := f(Z \cup \{v\}) - f(Z)$. Soit Z_k^* un maximiseur de f de taille k, et Z_k, la solution construite par l'algorithme 2, avec Z_i (pour $i < k$) les itérations successives depuis $Z_0 = \emptyset$. On note aussi, $Z_{i+1} = Z_i \cup \{v_{i+1}\}$.

Lemme 2.

$$\forall k > 0, f(Z_k^*) \leq \sum_{i=1}^{k} f(\{v_i^*\}),$$

où v_i^* sont les éléments de Z_k^* tels que $f(\{v_i^*\})$ est une suite décroissante.

Démonstration.

$$\begin{aligned} f(Z_k^*) &= \sum_{i=1}^{k} \Delta(v_i^* | \{v_1^*, \cdots, v_{i-1}^*\}) && \text{par téléscopage} \\ &\leq \sum_{i=1}^{k} \Delta(v_i^* | \emptyset) && \text{car } f \text{ sous-modulaire} \\ &= \sum_{i=1}^{k} f(\{v_i^*\}) \end{aligned}$$

□

Lemme 3.

$$\forall i, \forall v \notin Z_i, \Delta(v|Z_i) \geq w_i^T v,$$

Démonstration. Étant donné $n < N$, on nomme $z_i^{(n)}$ les éléments triés de $v_{[1:k]}^{(n)}$, tels que $z_1^{(n)} \geq \cdots \geq z_i^{(n)}$. f s'exprime comme :

$$f(Z_i) = \sum_{n=1}^{N} z_i^{(n)} \lambda^{i-1}$$

Pour calculer $f(Z_i \cup \{v\})$ pour n'importe quel v, pour chaque n, $v^{(n)}$ est inséré dans la séquence ordonnée des $z_i^{(n)}$. Soit $m(n)$ la position d'insertion. Autrement dit, $z_1^{(n)} \geq \cdots \geq z_{m(n)}^{(n)} \geq v^{(n)} \geq z_{m(n)+1}^{(n)} \geq \cdots \geq z_i^{(n)}$.

De même, on appelle $L^{(n)}$ et $R^{(n)}$ les parties gauche et droite de la somme :

$$L^{(n)} = \sum_{i \leq m(n)} z_i^{(n)} \lambda^{i-1}, \qquad R^{(n)} = \sum_{i > m(n)} z_i^{(n)} \lambda^{i-1}$$

Ainsi :

$$f(Z_i \cup \{v\}) = \sum_{n=1}^{N} L^{(n)} + \lambda^{m(n)} v^{(n)} + \lambda R^{(n)}.$$

$$\begin{aligned}\Delta(v|Z_i) &= f(Z_i \cup \{v\}) - f(Z_i) \\ &= \sum_{n=1}^{N} \left[L^{(n)} + \lambda^{m(n)} v^{(n)} + \lambda R^{(n)}\right] - \sum_{n=1}^{N} \left[L^{(n)} + R^{(n)}\right] \\ &= \sum_{n=1}^{N} \lambda^{m(n)} v^{(n)} - (1-\lambda) \sum_{n=1}^{N} R^{(n)}\end{aligned}$$

Maintenant, à propos de $w_{i-1}^T v$, on note $c(n)$ le nombre de $v_j^{(n)}$ strictement positifs ($1 \leq j < i$). Par construction, $w_{i-1}^{(n)} = \lambda^{c(n)}$, d'où :

$$w_{i-1}^T v = \sum_{n=1}^{N} \lambda^{c(n)} v^{(n)}.$$

$$\begin{aligned}&\Delta(v|Z_i) - w_{i-1}^T v \\ &= \sum_{n=1}^{N} \lambda^{m(n)} v^{(n)} - (1-\lambda) \sum_{n=1}^{N} R^{(n)} - \sum_{n=1}^{N} \lambda^{c(n)} v^{(n)} \\ &= \sum_{n=1}^{N} (\lambda^{m(n)} - \lambda^{c(n)}) v^{(n)} - (1-\lambda) \sum_{n=1}^{N} \sum_{i > m(n)} z_i^{(n)} \lambda^{i-1} \\ &\geq \sum_{n=1}^{N} (\lambda^{m(n)} - \lambda^{c(n)}) v^{(n)} - (1-\lambda) \sum_{n=1}^{N} \sum_{m(n) < i \leq c(n)} z_i^{(n)} \lambda^{i-1} && \text{par construction de } c(n) \\ &\geq \sum_{n=1}^{N} (\lambda^{m(n)} - \lambda^{c(n)}) v^{(n)} - \sum_{n=1}^{N} v^{(n)} (1-\lambda) \sum_{m(n) < i \leq c(n)} \lambda^{i-1} && \text{pour ces } i,\ z_i^{(n)} \leq v^{(n)} \\ &= \sum_{n=1}^{N} (\lambda^{m(n)} - \lambda^{c(n)}) v^{(n)} - \sum_{n=1}^{N} v^{(n)} (\lambda^{m(n)} - \lambda^{c(n)}) && \text{somme téléscopique} \\ &= 0\end{aligned}$$

□

Lemme 4.

$$\forall k > 0, f(Z_k) \geq \sum_{i=1}^{k} f(\{v_i^*\}) \lambda^{i-1},$$

où v_i^* sont éléments de Z_k^* tels que $f(\{v_i^*\})$ est une suite décroissante.

Démonstration.

$$\begin{aligned} f(Z_k) &= \sum_{i=1}^{k} \Delta(v_i|Z_{i-1}) \\ &\geq \sum_{i=1}^{k} w_{i-1}^T v_i \qquad \text{grâce au Lemme (3)} \end{aligned}$$

Soit $i \leq k$. Prouvons que $w_{i-1}^T v_i \geq \lambda^{i-1} f(\{v_i^*\}) = \lambda^{i-1} 1^T v_i^*$. Il y a deux cas :

- Si $v_i^* \notin Z_i$, alors par construction : $w_{i-1}^T v_i \geq w_{i-1}^T v_i^*$. Puisque chaque $w_{i-1}^{(n)} \geq \lambda^{i-1}$, alors $w_{i-1}^T v_i \geq \lambda^{i-1} 1^T v_i^*$.
- Si $v_i^* \in Z_i$, on sait que $\exists j < i$ tel que $v_j^* \notin Z_i$. Donc $w_{i-1}^T v_i \geq w_{i-1}^T v_j^* \geq \lambda^{i-1} 1^T v_j^*$. Puique $f(\{v_i^*\})$ suite décroissante, $1^T v_j^* \geq 1^T v_i^*$. Par conséquent, $w_{i-1}^T v_i \geq \lambda^{i-1} 1^T v_i^*$.

D'où, $f(Z_k) \geq \sum_{i=1}^{k} f(\{v_i^*\}) \lambda^{i-1}$. □

En combinant les Lemmes (2) et (4), on obtient :

$$\frac{f(Z_k)}{f(Z_k^*)} \geq \frac{\sum_{i=1}^{k} f(\{v_i^*\}) \lambda^{i-1}}{\sum_{i=1}^{k} f(\{v_i^*\})}.$$

Grâce à l'inégalité de Tchebychev pour les sommes :

$$\frac{f(Z_k)}{f(Z_k^*)} \geq \frac{\sum_{i=1}^{k} \lambda^{i-1}}{k} = \frac{1}{k} \frac{1-\lambda^k}{1-\lambda},$$

D'où la borne. □

Enfin, nous pouvons montrer que la borne est atteinte dans un scénario, signifiant qu'il s'agit bien de la meilleure borne.

Démonstration. Pour cela, nous supposons $N > k$.

On suppose que l'ensemble de candidats est $T = (z_1, \cdots, z_N, e_1, \cdots e_N)$, tel que

$$\begin{cases} \forall i, z_i = (\frac{1}{N}, \cdots, \frac{1}{N}) \\ \forall i, e_i^{(n)} = \mathbb{1}(i = n) \end{cases}$$

Si l'algorithme s'obstine à choisir seulement des z_j, alors $w_i = (\lambda^i, \cdots, \lambda^i)$. Ce faisant, $\forall z \in T, w_i^T z = \lambda^i$. Par conséquent, l'algorithme peut choisir de manière équivalente n'importe quel $z \in T$ pour la prochaine itération. Si il ne choisit que des z_j pour Z_k, alors :

$$f(Z_k) = 1 + \lambda + \cdots + \lambda^{k-1} = \frac{1-\lambda^k}{1-\lambda}$$

D'autre part, le meilleur ensemble Z_k^* de k éléments est constitué uniquement de e_j :

$$f(Z_k^*) = k$$

Le ratio est celui de l'équation (4). □

D Compléments

Pour avoir une vision complète des résultats, nous présentons les tableaux de couverture, pertinence et longueur (tableau 3), ainsi que des scores BLEU par domaine (tableau 4 et tableau 5).

Les domaines offrent une grande variabilité dont il est difficile d'extraire des tendances. Notons cependant que Europarl et Wikipedia ne bénéficient pas d'une plus grande couverture car ce sont les cas où $\lambda = 1$ qui donnent le meilleur BLEU. C'est le contraire pour ECB et JRC-Aquis.

	test-0.4			*test-0.6*		
recherche	couv.	perti.	long.	couv.	perti.	long.
SDM-0	70,8	36,7	35,6	77,4	46,0	31,3
SDM-0,2	71,0	36,5	36,1	77,5	45,7	31,7
SDM-1	66,7	38,1	37,8	75,2	47,7	32,1
SDM-IDF-0	70,4	36,3	34,8	77,2	45,5	31,4
SDM-IDF-0,2	70,8	36,4	35,0	77,4	45,5	31,5
SDM-IDF-1	68,5	37,7	37,6	75,3	46,9	32,4
NGM-0	70,3	38,5	30,5	77,4	47,2	28,3
NGM-0,2	70,3	39,1	31,0	77,4	48,1	28,6
NGM-1	66,6	40,4	32,4	74,6	49,7	29,1
NGM-IDF-0	70,4	38,3	30,3	77,5	46,9	28,1
NGM-IDF-0,2	70,6	38,9	30,8	77,5	47,7	28,4
NGM-IDF-1	66,9	40,2	32,3	74,8	49,1	29,3
DL-0	70,6	40,5	27,5	78,6	50,5	24,5
DL-0,2	71,2	39,0	31,7	78,9	48,7	28,0
DL-1	62,1	47,2	20,0	72,3	56,9	19,7
DL-IDF-0	70,6	39,5	28,1	78,6	49,5	25,1
DL-IDF-0,2	70,9	38,7	30,9	78,8	48,3	27,8
DL-IDF-1	62,1	47,2	20,0	72,3	56,9	19,7
DL-MMR	64,3	46,1	20,4	73,6	56,3	19,9

TABLE 3 – Couverture, pertinence et longueur moyennes selon le score de recherche choisi, avec ou sans normalisation IDF, et $\lambda \in \{0; 0{,}2; 1\}$.

	ECB	EME	Epp	JRC	Ubu	Wiki	moy
test-0.4							
SDM-0	55,0	54,1	33,8	66,8	45,7	32,5	48,0
SDM-0,2	54,7	54,4	33,2	66,8	45,4	32,2	47,8
SDM-1	55,1	53,1	34,3	66,9	45,0	33,1	47,9
SDM-IDF-0	55,1	54,5	33,5	67,1	45,9	32,0	48,0
SDM-IDF-0,2	55,0	54,4	33,6	67,0	45,4	32,3	47,9
SDM-IDF-1	55,5	53,4	33,1	67,3	44,7	33,2	47,9
NGM-0	56,1	56,6	34,7	68,2	46,0	33,6	49,2
NGM-0,2	56,2	56,2	34,9	68,3	45,8	33,9	49,2
NGM-1	56,0	55,7	35,3	67,8	45,3	33,8	49,0
NGM-IDF-0	55,7	56,4	34,4	68,0	45,5	33,5	48,9
NGM-IDF-0,2	56,0	56,5	34,7	68,1	45,4	33,8	49,1
NGM-IDF-1	55,8	55,6	34,9	67,8	45,7	34,7	49,1
DL-0	57,0	55,6	35,2	68,7	46,1	33,9	49,4
DL-0,2	56,6	55,6	35,0	68,4	45,7	33,8	49,2
DL-1	55,9	55,2	35,5	68,0	46,6	34,7	49,3
DL-IDF-0	56,1	55,5	34,5	68,1	46,1	33,5	49,0
DL-IDF-0,2	56,3	55,4	34,6	68,0	46,9	34,1	49,2
DL-IDF-1	55,9	55,2	35,4	68,0	46,7	34,8	49,3
test-0.6							
SDM-0	64,5	62,5	50,1	75,7	53,6	60,8	61,2
SDM-0,2	64,1	62,5	50,1	75,8	53,9	60,6	61,2
SDM-1	63,8	61,9	50,1	74,8	52,3	61,2	60,7
SDM-IDF-0	64,9	61,6	49,4	76,0	52,8	59,9	60,8
SDM-IDF-0,2	64,7	61,9	49,2	75,8	53,6	60,2	60,9
SDM-IDF-1	64,7	61,8	50,0	75,5	52,0	61,2	60,9
NGM-0	64,2	63,6	51,0	76,7	53,0	61,1	61,6
NGM-0,2	64,9	63,7	51,2	76,8	53,2	61,5	61,9
NGM-1	64,8	63,8	51,2	76,5	52,5	62,9	61,9
NGM-IDF-0	64,7	64,0	51,1	76,8	53,1	61,4	61,8
NGM-IDF-0,2	65,7	63,5	51,0	76,8	53,5	61,5	62,0
NGM-IDF-1	65,5	63,8	50,8	76,3	52,6	62,0	61,8
DL-0	65,4	65,5	50,6	77,1	54,2	62,0	62,5
DL-0,2	65,4	65,8	51,0	77,0	53,9	61,9	62,5
DL-1	64,9	65,1	51,2	76,8	53,3	62,4	62,3
DL-IDF-0	65,3	65,2	50,7	77,1	53,7	61,3	62,2
DL-IDF-0,2	65,7	64,9	51,0	76,9	53,9	61,2	62,3
DL-IDF-1	64,9	65,1	51,2	76,8	53,3	62,4	62,3

TABLE 4 – Score BLEU selon le score de recherche choisi, avec ou sans normalisation IDF, et $\lambda \in \{0; 0{,}2; 1\}$.

λ	ECB	EME	Epp	JRC	Ubu	Wiki	moy
test-0.4							
0	57,0	55,6	35,2	68,7	46,1	33,9	49,4
0,1	56,4	55,3	34,7	68,4	46,3	33,7	49,1
0,2	56,6	55,6	35,0	68,4	45,7	33,8	49,2
0,3	56,5	55,4	35,0	68,6	45,9	33,6	49,2
0,4	56,9	55,5	34,8	68,2	45,6	33,9	49,1
0,5	57,0	56,0	34,5	68,4	46,0	34,5	49,4
0,6	57,1	55,8	34,7	68,2	45,7	34,5	49,3
0,7	56,8	55,6	35,1	68,2	45,5	34,4	49,3
0,8	56,6	55,2	35,0	68,2	45,0	34,3	49,1
0,9	56,5	55,4	35,2	68,1	45,2	34,2	49,1
1	55,9	55,2	35,5	68,0	46,6	34,7	49,3
test-0.6							
0	65,4	65,5	50,6	77,1	54,2	62,0	62,5
0,1	65,5	65,4	50,8	77,0	53,8	61,8	62,4
0,2	65,4	65,8	51,0	77,0	53,9	61,9	62,5
0,3	65,6	65,5	51,2	77,1	53,8	62,0	62,5
0,4	65,9	65,3	51,2	77,1	54,2	62,1	62,6
0,5	66,0	65,2	51,2	77,2	54,4	62,1	62,7
0,6	66,0	65,0	51,1	77,1	54,2	62,4	62,6
0,7	65,7	64,7	51,1	76,9	54,1	62,3	62,5
0,8	65,6	65,1	50,8	76,8	53,7	62,4	62,4
0,9	65,5	65,2	50,8	76,8	53,5	62,4	62,3
1	64,9	65,1	51,2	76,8	53,3	62,4	62,3

TABLE 5 – Score BLEU selon la valeur de λ pour DL avec normalisation cardinale.