Aprentissage non-supervisé pour l'appariement et l'étiquetage de cas cliniques en français - DEFT2019

Damien Sileo[1, 2] Tim Van de Cruys[2, *] Philippe Muller[2] Camille Pradel[1]
(1) Synapse Développement, 5 Rue du Moulin Bayard, 31000 Toulouse
(2) IRIT, Université Paul Sabatier 118 Route de Narbonne 31062 Toulouse
damien.sileo@synapse-fr.com, camille.pradel@synapse-fr.com,
philippe.muller@irit.fr, tim.van-de-cruys@irit.fr

RÉSUMÉ
Nous présentons le système utilisé par l'équipe Synapse/IRIT dans la compétition DEFT2019 portant sur deux tâches liées à des cas cliniques rédigés en français : l'une d'appariement entre des cas cliniques et des discussions, l'autre d'extraction de mots-clefs. Une des particularité est l'emploi d'apprentissage non-supervisé sur les deux tâches, sur un corpus construit spécifiquement pour le domaine médical en français

ABSTRACT
Unsupervised learning for matching and labelling of french clincal cases - DEFT2019

We present the system used by the Synapse / IRIT team in the DEFT2019 competition covering two tasks on clinical cases written in French : the matching between clinical cases and discussions, and the extraction of key words. A particularity of our submissions is the use of unsupervised learning on both tasks, thanks to a french corpus of medical texts we gathered.

MOTS-CLÉS : TALN bio-médical, DEFT2019, apprentissage non-supervisé.

KEYWORDS: biomedical NLP, DEFT2019, unsupervised learning.

1 Introduction

Les textes du domaine médical sont une source d'information précieuse dont l'analyse automatique peut aider la recherche et le traitement des patients. Cependant, leur nature non structurée fait de leur analyse automatique est un défi, amplifié par la technicité et la spécificité du langage employé. Cette difficulté est exacerbée dans le cas du français, pour lequel les travaux et ressources sont plus rares.

La campagne d'évaluation DEFT2019 (Grabar *et al.*, 2019) est la première à porter sur des textes cliniques français. Les données sont constituées de cas cliniques, accompagnée de discussions correspondantes. Sur ces données, la campagne d'évaluation propose en outre les tâches suivantes :

— L'étiquetage de couples cas cliniques/discussions, par plusieurs expressions clefs choisies dans un ensemble pré-défini de 1311 expressions. (tâche 1)
— L'appariement de cas cliniques avec des discussions originellement correspondantes (tâche 2)

Une autre tâche portant sur l'extraction d'informations (e.g. âge, sexe des patients) n'est pas traitée ici. Dans cet article, on utilise des techniques d'apprentissage non-supervisé pour participer aux deux taches (particulièrement pour la tâche 2).

Pour ce faire, on construit un ensemble de corpus basé sur des ressources en français qui servira à

Jeux de données	Nombre de documents
EMEA	26289
Aranea-Med	11093
Wac-Med	2514
Cochrane	7676
Wiki-Med	4933
Deft	3974
Quaero	3479

TABLE 1 – Nombre de documents dans les jeux de données utilisés

pré-entrainer un réseau de neurones basé sur une concaténation d'embeddings différents, et d'un encodage des textes à base de ces représentations vectorielles de mots par des réseaux convolutifs.

2 Constitution d'un corpus de textes médicaux

Pour servir à l'apprentissage non-supervisé, on constitue un corpus médical à partir des sources suivantes :

— L'ensemble des textes de DEFT2019, y compris les données de test
— Les articles de Wikipédia appartenant au portail de la médecine, que nous nommerons Wiki-Med
— EMEA (Tiedemann, 2012) qui contient des textes de l'european medical agency
— Quaero (Névéol *et al.*, 2014), qui contient des titres Medline et des documents EMEA annotés, pour la reconnaissance d'entités nommées et la normalisation (ici, on ignore ces annotations)
— Des résumés d'articles Cochrane (Grabar & Cardon, 2018)

De plus, on augmente ce corpus en utilisant une technique simple d'adaptation de domaine. On entraine un classifieur FastText (Joulin *et al.*, 2016) (paramètres par défauts, 2 itérations) afin d'apprendre à prédire si des textes viennent du corpus Wiki-Med, ou de sources web (Aranea(Panchenko *et al.*, 2017), FrWac(Ferraresi *et al.*, 2008)) échantillonnées de sorte à contenir deux fois plus de textes que Wiki-Med.

À partir de ce classifieur, on extrait des corpus web les textes qui sont prédits comme appartenant à Wiki-Med. Ces textes sont présents en plus grand nombre que ceux de Wiki-Med. Si le classifieur s'est mépris sur leur origine, c'est, on l'espère, qu'ils sont lexicalement proches de ceux de Wiki-Med, autrement dit qu'ils concernent la médecine, et qu'ils seront utiles pour l'apprentissage de représentations de mots ou d'encodeurs de textes. L'aggrégation de ces corpus, ayant subie une déduplication des textes strictement identiques sera nommé Fr-Med dans la suite de l'article. Les nombres de documents selon ces différentes sources sont donnés dans la table 2.

3 Prétraitement du texte

Les textes passent par la fonction *fix_text* de la librairie ftfy (Speer, 2019) afin de remédier à certains problèmes d'encodage. Ils sont ensuite passés en minuscules. Les nombres entre crochets (citations) sont éliminés, de même que les chiffres entre parenthèses. Les virgules et les mots vides de la liste

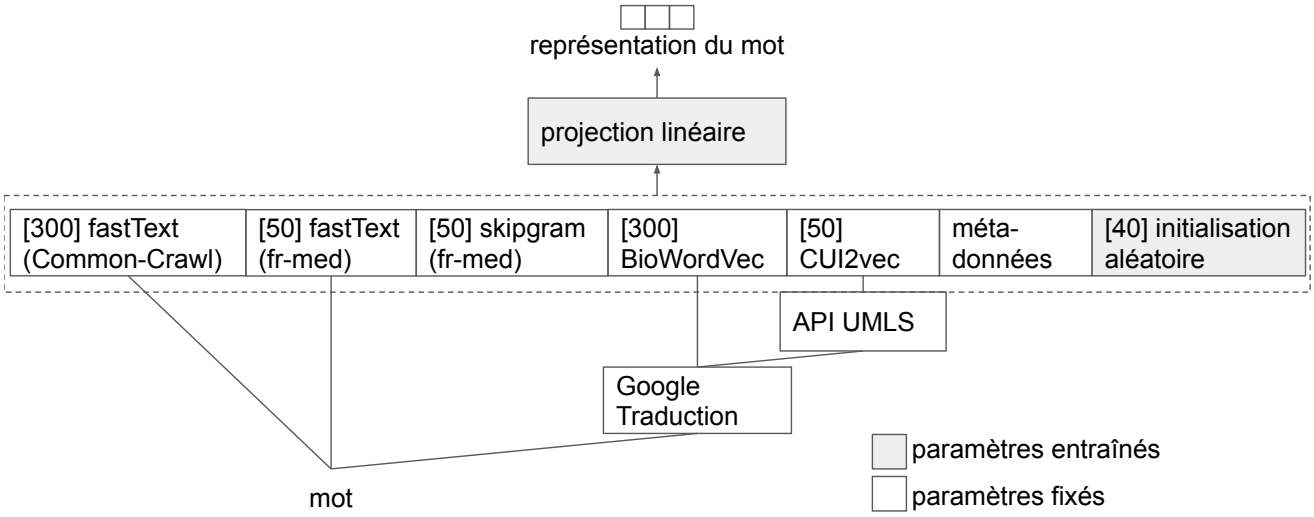

FIGURE 1 – Représentation vectorielle d'un mot. Les chiffres entre crochets désignent la dimension

stopwords-iso [1] sont éliminés.

4 Embeddings

Les embeddings utilisés sont les suivants, aussi représentés dans la figure 1 :

— FastText (Bojanowski *et al.*, 2017) pré-entraînés sur CommonCrawl distribuées sur `fasttext.cc` [2]

— Des embeddings FastText appris sur le corpus *Fr-Med* décrit précédemment, avec une taille de 50, 12 epochs et les paramètres par défaut sinon

— Des embeddings SkipGram appris sur le corpus *Fr-Med* et accédés par PyMagnitude (Patel *et al.*, 2018)

— BioWord2Vec (Chen *et al.*, 2018) [3] utilisés à la suite d'une traduction en anglais utilisant google API [4]

— CUI2Vec (Beam *et al.*, 2018) qui sont des embeddings de Concept Unique Identifier (CUI) UMLS. Le lien entre les mots et les CUI est obtenu en utilisant la fonction de recherche l'API publique UMLS. Leur dimension a été réduite à 50 en utilisant (Raunak, 2017).

— Des méta-données : l'appartenance aux dictionnaires de chaque embeddings (5 booléens), la fréquence d'occurence dans les documents de DEFT (répartie en 12 quantifications binaires)

— Une partie initalisée aléatoirement et apprise lors de l'optimisation des taches finales, de dimension 40

5 Tâche 2 - Appariement des cas cliniques et des discussions

5.1 Modélisation du problème

On traite le problème comme de la classification à partir des paires de phrases, la classe prédite étant l'existence d'un lien entre un cas et une discussion. Les données d'entrainement fournissent déjà

1. https ://github.com/stopwords-iso/stopwords-iso
2. `https://fasttext.cc/docs/en/crawl-vectors.html`
3. `https://ftp.ncbi.nlm.nih.gov/pub/lu/Suppl/BioSentVec/BioWordVec_PubMed_MIMICIII_d200.bin`
4. Les API ont été consommées en mai 2019

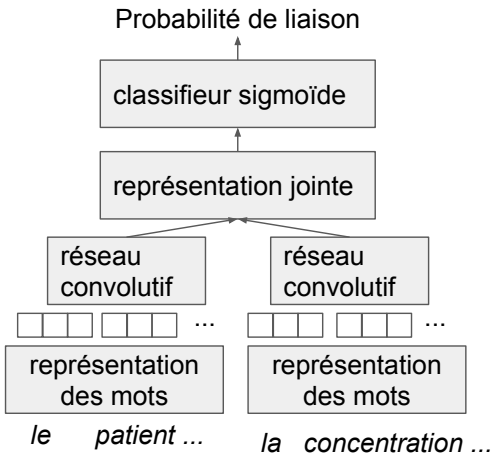

FIGURE 2 – Architecture de la prédiction de liaison entre cas et discussion

des cas et discussions liés. Pour générer des exemples non-liés, on applique un produit cartésien entre les cas et les discussions (en éliminant les cas et dicussions liées). Ensuite, les paires liées sont suréchantillonées par un facteur 10 afin de diminuer le déséquilibre des classes, sans pour autant perdre des données en sous-échantillonnant les exemples non-liés. 15% des données (brutes, c'est à dire avant le produit cartésien) sont réservées à la validation.

La figure 2 montre l'architecture du système utilisé pour la tache 2. Un réseau convolutif compose les représentations de mots décrites précedemment afin de représenter les cas et discussions par un vecteur de taille fixe. Ces vecteurs sont eux mêmes composés en une représentation jointe qui sert à classifier la présence de lien entre cas et discussion. La représentation jointe est $\text{ReLu}(W[u, v, u \odot v, |u - v + t|)]$ (Sileo *et al.*, 2019) où u et v sont les sorties des réseaux convolutifs t est un paramètre de la même dimension que u et v. Les paramètres de ce réseau sont optimisés de sorte à minimiser l'entropie croisée. Les probabilités de laison obtenues permettent de classer pour chaque cas l'ensemble des discussions selon leur pobabilité, et la plus probable est prédite dans les soumissions. La métrique d'évaluation est la proportion de cas pour lequels la bonne discussion a été trouvée, qu'on nomme précision.

5.2 Représentation des séquences de mots

Le réseau convolutif est constitué de $N = 768$ filtres de taille 1, et $N = 768$ filtres de taille 3, concaténés et suivis par une activation ReLu et d'un max-pooling. La taille des séquences de mots d'entrée est limitée à 600.

5.3 Hyperparamètres

On utilise l'optimiseur Adam (Kingma & Ba, 2014) avec le learning rate 0.002, déterminé par validation croisée.

5.4 Pré-entrainement pour l'appariement

On pré-entraine le réseau convolutif et la représentation de mots en utilisant une tâche d'apprentissage non-supervisée inspirée de (Devlin *et al.*, 2018) et (Logeswaran & Lee, 2018) qui consiste découper chaque document du corpus Fr-Med, en deux parties, puis à prédire si deux parties appartiennent au même document. On ne garde que les documents d'au moins 40 mots. On pourrait simplement

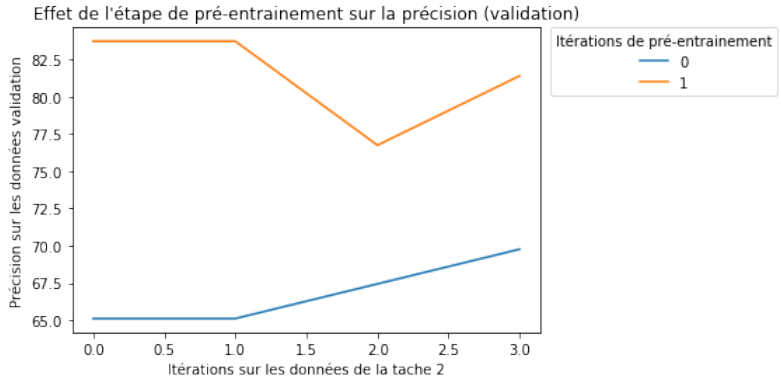

FIGURE 3 – Courbe d'apprentissage pour la tache 2

découper en deux parties égales les documents, mais dès lors le réseau pourrait apprendre à apparier les séquences de même taille, ce qui n'est pas intéressant pour l'apprentissage de représentations. On choisit donc cette division : l'endroit qui divise les deux parties des documents est choisi selon une distrbution de probabilité uniforme telle que la taille de chacun des segments soit supérieure à 20.

Les exemples négatifs sont générés en appariant aléatoirement des segments n'appartenant pas au même document. Les segments appartenant au même document étant a priori thématiquement proches, du moins plus proches en moyenne que des segments issus d'autres documents pris au hasard, cette tâche permet de tirer parti de Fr-Med pour entrainer les encodeurs de textes.

Le jeu de données résultat contient $1.4M$ exemples dont 10% de segments appartenant au même document.

5.5 Influence du pré-entrainement

La figure 3 montre l'influence de cette étape pré-entrainement. Sans l'itération de pré-apprentissage, la précision reste limitée même après plusieurs itérations sur les données de la tâche 2. La tache de pré-entrainement semble donc assez liée à la tache 2 pour être utile.

5.6 Sélection et aggrégation de modèles

On entraine plusieurs modèles, sur des ensemble d'entrainement et de validation distincts, et on selectionne ceux qui ont les meilleures performances sur les données de validation. Les probabilités de liaisons des différents modèles sont aggrégées par une moyenne.

5.7 Résultats

Les résultats des différents runs figurent dans la table 5.7. Le run 2 est issu d'un seul modèle, et les runs 1 et 3 utilisent 4 et 6 modèles.

6 Tâche 1 - Étiquetage des cas cliniques

6.1 Apprentissage non-supervisé pour la détection de mots-clefs

On génère un corpus d'apprentissage non-supervisé pour l'étiquetage de textes avec les données web Aranea(Panchenko *et al.*, 2017). Pour ce faire, on parcourt les pages, et lorsqu'un mot-clef de la liste prédéfinie par la tâche apparait, on le considère comme un label à prédire. Les labels à

	précision (test entier)	précision (test dédupliqué)
run 1	57.94%	61.68%
run 2	54.21%	56.07%
run 3	57.00%	63.08%

TABLE 2 – Résultats sur les données de test de la tache 2 (précision), avec les exemples de tests dans leur totalité, et sur un ensemble exempt de doublons

prédire sont masqués dans 90% des cas pour que le modèle n'apprenne pas une simple détection de la présence stricte des mots clefs. Le corpus résultant contient $16.8k$ exemples contenant au moins des mots-clefs. Le réseau de convolution défini dans la section 5.2, mais avec une seule convolution de taille 1 avec 256 filtres, suivi d'un classifieur log-linéaire, est entrainé de sorte à prédire les mots clefs, Le modèle est pré-entrainé à la prédiction de ces mots-clefs sur ce corpus sur 8 itérations. Le modèle est ensuite optimisé pour la même tâche sur les données de DEFT2019 (1874 exemples) (train/test, cas/discussions) dédupliquées, avec 3 itérations. Notons bien qu'il s'agit de prédiction de mots clefs présents dans le textes et non pas de mots clefs étant ceux qui sont des labels à prédire, fixés par l'annotation mise en oeuvre pour la tâche.

6.2 Modèle de classification

Soit n le nombre de labels (ici 1311) Pour chaque label, on calcule K traits par exemple. Les traits utilisés sont les suivants :

- présence du mot-clef dans un texte $x_i = 1$ si le label i est dans le texte présent, $x_i = 0$ sinon.
- présence du mot-clef sans qu'un autre mot-clef plus petit soit présent dans le texte
- présence prédite par le classifieur de mots clefs décrit dans la sous-section précédente, élevée aux puissances 1,2 et 4.
- score du mot clef selon la librairie fuzzywuzzy [5] qui détecte des mots clefs ou des éditions de leur chaine de caractères en se basant sur des distances de levenstein. La fonction *p*artial_score est utilisée. Les scores sont seulement comptabilisés s'ils dépassent le seuil de 0.85.

Un paramètre θ pondère ces K traits, pour fournir un score pour chaque mot clef dans le cas ou dans la discussion. Les paramètres θ et α sont intialisés de sorte à ce que chaque composante vaille 1. Les scores du cas et de la discussion sont agrégés avec une moyenne pondérée : $\text{score}(\text{exemple}) = \text{score}(\text{discussion}) + a.\text{score}(\text{cas})$ Les paramètres de ce réseau sont optimisés de sorte à minimiser l'entropie croisée.

6.3 Hyperparamètres

La valeur déterminée $a = 0.3$, ce qui signifie que la discussion semble plus utile que le cas pour déterminer les mots clefs. On utilise l'optimiseur Adam (Kingma & Ba, 2014) avec le learning rate 0.01. Une régularisation d'activité L1 est appliquée à la sortie du réseau, avec un coefficient $\lambda = 10^{-5}$ (ce qui revient à ajouter $\lambda|y|_{L1}$ à la fonction de coût). Ces hyperparamètres ont été choisis par validation croisée.

6.4 Résultats

Les résultats des différents runs figurent dans la table 6.4 On entraine une multitude de modèles et les selectionne selon leur précision sur des ensembles de validations différents à chaque fois (la

5. https://github.com/seatgeek/fuzzywuzzy

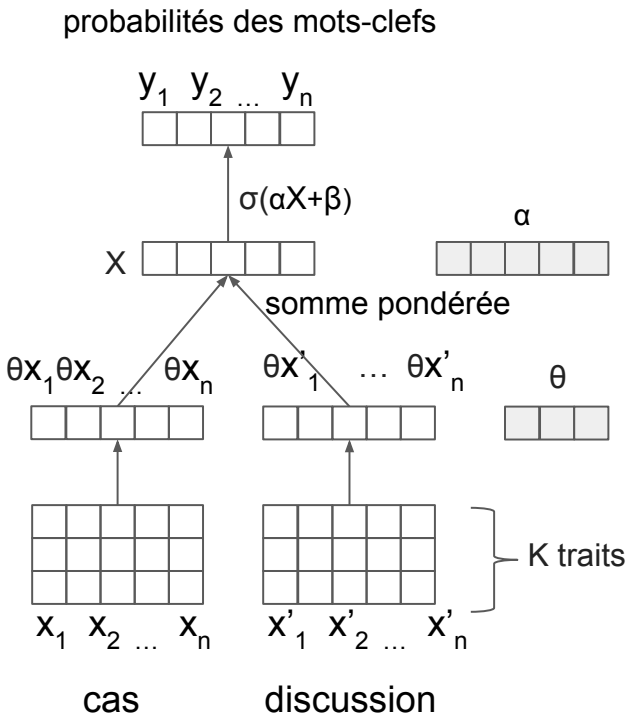

FIGURE 4 – Architecture tache 1. θ et α sont des paramètres libres. σ dénote la fonction sigmoïde. Les vecteurs en gris sont des paramètres appris

	MAP	P@n
run 1	36.47%	43.87%
run 2	44.64%	43.90%
run 3	36.53%	43.87%

TABLE 3 – Résultats sur les données de test de la tache 1

.

partition étant aléatoire, avec 85% de données pour l'entrainement) Le run 1 est un ensemble de 13 modèles ayant une précision supérieure à 29% en validation. Le run 2 est un ensemble de 8 modèles ayant une précision supérieure à 26% sur les données de validation. Le run 3 est un ensemble de 12 modèles ayant une précision supérieure à 30% en validation. Visiblement, la selection de modèles par la précision en validation n'est pas optimale.

7 Conclusion

On a décrit deux systèmes, faisant appel à de l'apprentissage non-supervisé, pour l'étiquetage de cas cliniques et l'appariement avec des discussions. Il serait intéressant d'utiliser des techniques des traduction de manière plus poussée afin de pouvoir bénéficier des techniques récentes d'apprentissage non-supervisé (Devlin *et al.*, 2018) et leurs déclinaisons pour le domaine biomédical(Lee *et al.*, 2019).

Pour la tâche 2, il serait intéressant d'évaluer un modèle non pas simplement pré-entrainé mais multi-tâches entrainé à la fois à la tache 2 et à la tache de pré-entrainement, pour atténuer l'oubli de cette dernière (Kirkpatrick *et al.*, 2016).

Les résultats de la tache 1 pourraient être améliorés en faisant appel à des techniques de recherche d'information plus poussée (e.g. utilisation d'ElasticSearch) pour la création d'autres traits.

Enfin, les deux tâches étant liées à de l'ordonnancement, des fonctions de coûts plus appropriées pourraient être considérées.

Références

BEAM A. L., KOMPA B., FRIED I., PALMER N. P., SHI X., CAI T. & KOHANE I. S. (2018). Clinical concept embeddings learned from massive sources of medical data. *CoRR*, **abs/1804.01486**.

BOJANOWSKI P., GRAVE E., JOULIN A. & MIKOLOV T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association of Computational Linguistics*, **5**, 135–146.

CHEN Q., PENG Y. & LU Z. (2018). Biosentvec : creating sentence embeddings for biomedical texts.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2018). Bert : Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv :1810.04805*.

FERRARESI A., ZANCHETTA E., BERNARDINI S. & BARONI M. (2008). Introducing and evaluating ukwac, a very large web-derived corpus of english.

GRABAR N. & CARDON R. (2018). CLEAR-Simple Corpus for Medical French. In *ATA*, Tilburg, Netherlands.

GRABAR N., GROUIN C., HAMON T. & CLAVEAU V. (2019). Recherche et extraction d'information dans des cas cliniques. présentation de la campagne d'évaluation deft 2019. *Actes de DEFT*.

JOULIN A., GRAVE E., BOJANOWSKI P. & MIKOLOV T. (2016). Bag of Tricks for Efficient Text Classification.

KINGMA D. & BA J. (2014). Adam : A Method for Stochastic Optimization. *International Conference on Learning Representations*, p. 1–13.

KIRKPATRICK J., PASCANU R., RABINOWITZ N., VENESS J., DESJARDINS G., RUSU A. A., MILAN K., QUAN J., RAMALHO T., GRABSKA-BARWINSKA A., HASSABIS D., CLOPATH C., KUMARAN D. & HADSELL R. (2016). Overcoming catastrophic forgetting in neural networks. cite arxiv :1612.00796.

LEE J., YOON W., KIM S., KIM D., KIM S., SO C. H. & KANG J. (2019). Biobert : a pre-trained biomedical language representation model for biomedical text mining. *arXiv preprint arXiv :1901.08746*.

LOGESWARAN L. & LEE H. (2018). An efficient framework for learning sentence representations. p. 1–16.

NÉVÉOL A., GROUIN C., LEIXA J., ROSSET S. & ZWEIGENBAUM P. (2014). The QUAERO French medical corpus : A ressource for medical entity recognition and normalization. In *Proc of BioTextMining Work*, p. 24–30.

PANCHENKO A., RUPPERT E., FARALLI S., PONZETTO S. P. & BIEMANN C. (2017). Building a Web-Scale Dependency-Parsed Corpus from Common Crawl. p. 1816–1823.

PATEL A., SANDS A., CALLISON-BURCH C. & APIDIANAKI M. (2018). Magnitude : A fast, efficient universal vector embedding utility package. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing : System Demonstrations*, p. 120–126.

RAUNAK V. (2017). Simple and effective dimensionality reduction for word embeddings.

SILEO D., DE CRUYS T. V., PRADEL C. & MULLER P. (2019). Composition of sentence embeddings : Lessons from statistical relational learning. *CoRR*, **abs/1904.02464**.

SPEER R. (2019). ftfy. Zenodo. Version 5.5.

TIEDEMANN J. (2012). Parallel data, tools and interfaces in opus. In N. C. C. CHAIR), K. CHOUKRI, T. DECLERCK, M. U. DOGAN, B. MAEGAARD, J. MARIANI, J. ODIJK & S. PIPERIDIS, Eds., *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey : European Language Resources Association (ELRA).