UNIVERSITY
OF TECHNOLOGY
SYDNEY

Improving Low-Resource Named-Entity Recognition and Neural Machine Translation

by Iñigo Jauregi Unanue

Thesis submitted in fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

under the supervision of Prof. Massimo Piccardi

University of Technology Sydney

Faculty of Engineering and Information Technology

JUNE 2020

This thesis is dedicated to

my *family*.

Acknowledgments

I want to say thank you to my supervisor and friend Prof. Massimo Piccardi, for his brilliant guidance during my PhD. He has always trusted in my capacity as a researcher even though I initially had very limited experience in the field. He has also instilled in me passion for the field we are working in. Finally, he has taught me how to persevere in order to pursue good ideas and achieve my goals. Working with him has been an absolute pleasure.

Nire familiari ere eskerrak eman nahi dizkiot. Aita eta amari, beti nire ondoan egon direlako babes eta maitasun amaigabearekin. Bizitzari modu positiboan eta alai ekiten irakatsi didate eta beti babestu dituzte nire erabakiak. Beraiei esker naiz egun naizen pertsona. Iratiri ere, nire arrebari, eskerrak eman nahi dizkiot. Inork eduki dezaken arrebarik onena da, esan dezaket nire lagunik hoberena dela, eta zoragarria dela elkar zein ondo ulertzen garen. Urte hauetan beti kontatu izan dizkiot nire tesiaren gorabeherak eta beti egon da hor, entzuteko eta laguntzeko prest.

I would also like to dedicate a few, but heartfelt, words to my girlfriend, Nadia. She has been next to me every single day of my thesis, helping and supporting me in moments of despair and sharing the moments of happiness and success. I wouldn't have been able to finish this thesis without her. And to her family, specially to her parents, who have treated me as their own from the first time I came to this country.

I don't want to forget about my friend and colleague Ehsan. He was the one that gave me my first industry work opportunity in Australia, and since then, he has been great research advisor.

Finally, I want to mention that this research has been funded by the Rozetta Institute (formerly known as Capital Markets Cooperative Research Center).

Inigo Jauregi Unanue

June 2020, Sydney

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Inigo Jauregi Unanue declare that this thesis, is submitted in fulfilment of the requirements for the award of the degree of Doctor of Philosophy, in the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note: Signature removed prior to publication.

Date: 26.06.2020

Abstract

Named-entity Recognition (NER) and machine translation (MT) are two very popular and widespread tasks in natural language processing (NLP). The former aims to identify mentions of pre-defined classes (e.g. person name, location, time...) in text. The latter is more complex, as it involves translating text from a *source* language into a *target* language.

In recent years, both tasks have been dominated by deep neural networks, which have achieved higher accuracy compared to other traditional machine learning models. However, this is not invariably true. Neural networks often require large human-annotated training datasets to learn the tasks and perform optimally. Such datasets are not always available, as annotating data is often time-consuming and expensive. When human-annotated data are scarce (e.g. low-resource languages, very specific domains), deep neural models suffer from the overfitting problem and perform poorly on new, unseen data. In these cases, traditional machine learning models may still outperform neural models.

The focus of this research has been to develop deep learning models that suffer less from overfitting and can generalize better in NER and MT tasks, particularly when they are trained with small labelled datasets. The main findings and contributions of this thesis are the following. First, health-domain word embeddings have been used for health-domain NER tasks such as drug name recognition and clinical concept extraction. The word embeddings have been pretrained over medical domain texts and used as initialization of the input features of a recurrent neural network. Our neural models trained with such embeddings have outperformed previously proposed, traditional machine learning models over small, dedicated datasets. Second, the first systematic comparison of statistical MT and neural MT models over English-Basque, a low-resource language pair, has been conducted. This has shown that statistical models can perform slightly better than the neural models over the available datasets. Third, we have proposed a novel regularization technique for MT, based on regressing word and sentence embeddings. The regularizer has helped to considerably improve the translation quality of strong neural machine translation baselines. Fourth, we have proposed using reinforcement-style training with discourse rewards to improve the performance of document-level neural machine translation models. The proposed training has helped to improve the discourse properties of the translated documents such as the lexical cohesion and coherence over various low- and high-resource language pairs. Finally, a shared attention mechanism has helped to improve translation accuracy and the interpretability of the models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Natural language processing (NLP) is a field of research that combines linguistics, machine learning and computer science for the automatic analysis and synthesis of text. The main goal is to achieve "human-like language processing" in order to be able to develop intelligent systems that can act upon the information processed (e.g. clustering, classification, text generation). NLP is one of the key areas of artificial intelligence (AI) [Liddy, 2001].

In order to reach "human-like language processing", over the years the field has developed a wide range of sub-tasks. These sub-tasks or sub-fields of NLP process different aspects of language. Some of them are low-level NLP tasks, which analyze the lexical, gramatical and semantic elements of textual sentences and include text tokenization [Mikheev, 2003], part-of-speech tagging [Màrquez and Rodríguez, 1998], chunking [Ramshaw and Marcus, 1999], syntactic structure analysis [Socher *et al.*, 2011], named-entity recognition [Rau, 1991; Nadeau and Sekine, 2007], entity relation extraction [Mintz *et al.*, 2009], optical character recognition [Mori *et al.*, 1999] and speech processing (speech-to-text) [Hinton *et al.*, 2012] among others. Other tasks require a much higher level of abstraction to detect patterns that are implicit in text (closer to what we perceive as "AI"). High-level tasks include sentiment analysis [Pang *et al.*, 2008], natural language inference [Bowman *et al.*, 2015], fake-news detection [Shu *et al.*, 2017], deception detection [Mihalcea and Strapparava, 2009], machine translation [Koehn *et al.*, 2003; Bahdanau *et al.*, 2015], text summarization [Barzilay and Elhadad, 1999], poetry generation [Zhang and Lapata, 2014] and dialogue systems [Weizenbaum, 1966; Gao *et al.*, 2019].

And how does NLP address all of these tasks? What are the underlying algorithms? The early NLP systems (1950-1980s) relied heavily on human-designed rules and templates that were hard-coded into computer programs. These rules were often derived from well-established linguistic theories (e.g. generative grammar [Chomsky, 1956]), lexical analysis and domain knowledge. However, researchers soon became aware that these approaches were not scalable.

Therefore, later in the 80s, the field started to slowly move away from rules and embrace approaches based on machine learning (ML), such as decision trees and other probabilistic models. Researchers discovered that ML models could identify useful textual patterns from the data which enabled the rapid development of NLP systems across a wider range of domains.

Thanks to the fast progress in machine learning and the increased processing capability of computers, NLP research has now moved to the "deep learning era" [LeCun *et al.*, 2015]. Traditional machine learning models have been outperformed by large neural networks that have the ability to fit more complex datasets. These models have improved the accuracy of almost every existing NLP task, and thus, most of the current state-of-the-art NLP systems are underpinned by deep learning. Other areas of AI, such as computer vision, robotics and data mining, are also dominated by deep learning models.

Low-resource NLP

Despite the recent progress, deep neural networks also have their limitations. One of their main shortcomings is the *overfitting* problem [Lawrence *et al.*, 1997]. Overfitting occurs when a ML model learns to fit the training data "too well", and subsequently loses the ability to make accurate predictions over new, unseen data. This is common in ML models when the size of the training data is too small, yet it seems to be even more prevalent in deep neural networks (certainly, because of their much larger number of parameters). As can be observed in Figure 1.1, neural networks are "data hungry". That is, when there are enough training data these models perform the best, but accuracy decreases when the amount of data is limited (often below the standard of traditional machine learning models).

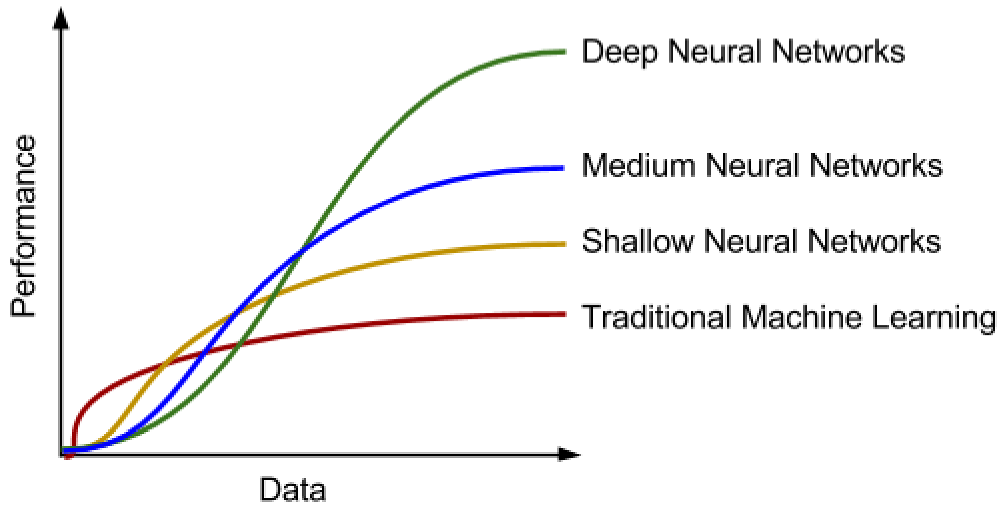


Figure 1.1: Performance vs data: Usual performance of different machine learning and deep learning models. Source: https://blog.easysol.net/building-ai-applications/

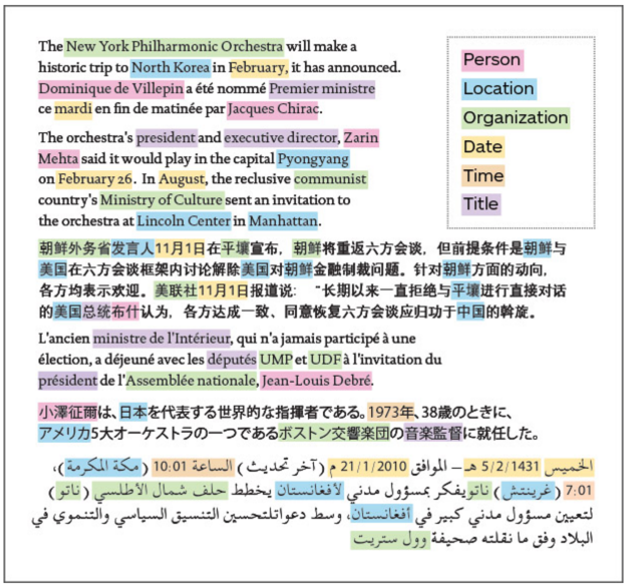
Consequently, overfitting poses a problem for NLP when the amount of human-annotated data is scarce. These include tasks with low-resource languages (e.g. Aboriginal languages in Australia) or specific language domains (e.g. the health domain, where data privacy issues prevail).

In the light of this, there is a clear need to improve the current deep learning models so that we can successfully use smaller training datasets without compromising the level of performance. Such improvements will contribute to the "democratization" of NLP by giving low-resource language speakers access to more accurate NLP tools. The aim of the research presented in this thesis is to address the overfitting problem in two specific NLP tasks: named-entity recognition (NER) and machine translation (MT).

Named Entity-Recognition and Machine Translation

NER seeks to identify mentions of predefined classes in unstructured text. It is a key step in many NLP applications such as information extraction, entity relation extraction and document de-identification. Typical pre-defined entities include person names, locations, organizations, time, currency, numbers and urls (amongst others). Figure 1.2 shows an example of a multilingual NER tagger.

On the other hand, MT is a task where the system needs to learn to translate a sentence



The New York Philharmonic Orchestra will make a historic trip to North Korea in February, it has announced.
Dominique de Villepin a été nommé Premier ministre ce mardi en fin de matinée par Jacques Chirac.

The orchestra's president and executive director, Zarin Mehta said it would play in the capital Pyongyang on February 26. In August, the reclusive communist country's Ministry of Culture sent an invitation to the orchestra at Lincoln Center in Manhattan.

朝鲜外务省发言人11月1日在平壤宣布，朝鲜将重返六方会谈，但前提条件是朝鲜与美国在六方会谈框架内讨论解除美国对朝鲜金融制裁问题。针对朝鲜方面的动向，各方均表示欢迎。美联社11月1日报道说："长期以来一直拒绝与平壤进行直接对话的美国总统布什认为，各方达成一致、同意恢复六方会谈应归功于中国的斡旋。

L'ancien ministre de l'Intérieur, qui n'a jamais participé à une élection, a déjeuné avec les députés UMP et UDF à l'invitation du président de l'Assemblée nationale, Jean-Louis Debré.

小澤征爾は、日本を代表する世界的な指揮者である。1973年、38歳のときに、アメリカ5大オーケストラの一つであるボストン交響楽団の音楽監督に就任した。

الخميس 5/2/1431 هـ — الموافق 21/1/2010 م (آخر تحديث) الساعة 10:01 (مكة المكرمة)، 7:01 (غرينتش) ناتويفكر بمسؤول مدني لأفغانستان يخطط حلف شمال الأطلسي (ناتو) لتعيين مسؤول مدني كبير في أفغانستان، وسط دعواتلتحسين التنسيق السياسي والتنموي في البلاد وفق ما نقلته صحيفة وول ستريت

Figure 1.2: An example of a commercial multilingual NER system. Source: https://www.basistech.com/text-analytics/rosette/entity-extractor/

from a *source* language to a *target* language. The translated sentence needs to retain the meaning of the original sentence and be grammatically correct. For example, the Spanish sentence *Hola, mi nombre es Iñigo*, should be translated into the English sentence *Hello, my name is Iñigo*. MT is more challenging than NER for two main reasons: 1) the length of the input and output sequences in MT can differ, and 2) the output space in MT is much larger compared to NER (whole vocabulary of the target language in MT vs few entity classes in NER).

Both state-of-the-art NER [Huang *et al.*, 2015; Lample *et al.*, 2016; Peters *et al.*, 2018; Devlin *et al.*, 2019] and MT [Bahdanau *et al.*, 2015; Sutskever *et al.*, 2014; Vaswani *et al.*, 2017] models are based on deep neural networks, and thus share the common problem of overfitting when the datasets are small. As mentioned above, the rest of the thesis will describe the research this author has carried out in order to alleviate this problem in both tasks.

1.1 Research Contributions

The thesis presents six main research contributions:

1. Pre-training of health-domain word embeddings for neural NER models in drug

name recognition and clinical concept extraction, which are two low resource NER tasks due to the limited available training data. Experimental results have shown that using our proposed health-domain embeddings together with general domain embeddings as input features provides a much better initialization to the neural network and helps to find a better local optimum during training with small datasets. Our approach has outperformed previously published results on two popular datasets.

2. Systematic comparison of three English-Basque MT models: a phrase-based statistical machine translation (PB-SMT) model trained by ourselves, a neural machine translation (NMT) model trained by ourselves, and a commercial off-the-shelf NMT system. Experiments over three low-resource datasets in this language pair have shown that PB-SMT models can outperform NMT models when the amount of training data available is limited. Moreover, we have seen that translating from English to Basque is more difficult than the other way round, probably because Basque is an agglutinative language, morphologically richer than English.

3. Proposal of a novel regularization technique for NMT models based on regressing word and sentence embeddings (ReWE and ReSE). Experiments over four different language pairs (with low-resource datasets) have shown that our regularization approach outperforms strong NMT baselines.

4. A detailed analysis was conducted to better understand how the proposed regularization methods (ReWE and ReSE) have achieved such significant improvements in translation accuracy. Our 2-D visualizations and clustering experiments over the output vectors have shown that the regularizers help to better discriminate between candidate translation words.

5. Proposal of a reinforcement-style training objective, leveraging discourse rewards, to improve the translation quality of document-level NMT models. There is a scarcity of document-level MT training datasets, and thus, NMT models struggle to learn to generate accurate translations with good discourse structure. The proposed training method alleviates this problem by explicitly targeting the maximization of two established discourse metrics such as lexical cohesion(LC) and

coherence(COH).

6. Proposal of a novel shared attention mechanism for neural automatic post-editing (APE) model. APE models are generally trained over small training datasets in order to learn to correct the errors made by a black-box MT system. The proposed shared attention has let the model select from words in the source sentence or the black-box MT output and helps to improve the translation accuracy of strong APE baselines. Moreover, the attention has shown to be useful for a better interpretation of the results provided by the neural model.

1.2 Publications

This sections details the list of journal, conference and workshop publications completed during the completion of this PhD, as well as the presentations given by the author.

- Journal papers
  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi and Massimo Piccardi, "Recurrent Neural Networks with Specialized Word Embeddings for Health-Domain Named-Entity Recognition," Journal of Biomedical Informatics, Vol. 72, pp. 102-109, December, 2017.
  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi and Massimo Piccardi, "Regressing Word and Sentence Embeddings for Regularization of Neural Machine Translation," (Under review on the Transaction of Neural Networks and Learning Systems (T-NNLS)).
- Conference papers
  - Iñigo Jauregi Unanue, Lierni Garmendia Arratibel, Ehsan Zare Borzeshi and Massimo Piccardi, "English-Basque Statistical and Neural Machine Translation", 11^{th} edition of the Language Resources and Evaluation Conference (LREC 2018), Miyazaki, Japan, 7-12 May, 2018, pp. 880-885.

  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi and Massimo Piccardi, "A Shared Attention Mechanism for Interpretation of Neural Automatic Post-Editing Systems", 2^{nd} Workshop on Neural Machine Translation and Generation (WNMT 2018), held in conjunction with ACL 2018, Melbourne, Australia, 20th of July, 2018, pp. 11-17.
  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi, Nazanin Esmaili and Massimo Piccardi, "ReWE: Regressing Word Embeddings for Regularization of Neural Machine Translation Systems", 11^{th} North American Chapter of the Association for Computational Lingusitics (NAACL-HLT 2019), Minneapolis, USA, 2-7 June, 2019, pp. 430-436.
  - Binh Thanh Kieu, Iñigo Jauregi Unanue, Son Bao Pham, Hieu Xuan Phan and Massimo Piccardi, "Learning Neural Textual Representations for Citation Recommendation", accepted at the International Conference on Pattern Recognition (ICPR) 2020, 8 pages.
  - Iñigo Jauregi Unanue, Nazanin Esmaili, Gholamreza Haffari and Massimo Piccardi, "Leveraging Discourse Rewards for Document-Level Neural Machine Translation," (Under review on an international NLP conference - ERA Rank A).

- Presentations
  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi and Massimo Piccardi, "Named-Entity Recognition (NER) in Health & Sentence Similarity", Emerging Trends in Digital Health (CMCRC 2017 annual conference, poster presentation), Sydney, Australia, 17-18 May, 2017.
  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi and Massimo Piccardi, "English-Basque Statistical and Neural Machine Translation", 15^{th} Annual Workshop of The Australian Language Technology Association (ALTA), Queensland University of Technology, Brisbane, Australia, 6-8 December, 2017.
  - Iñigo Jauregi Unanue, Ehsan Zare Borzeshi and Massimo Piccardi, "Health-Domain Natural Language Processing", Weekly Seminar of the IXA NLP Group, Donostia, Basque Country, Spain, 20th of December, 2017.

– Iñigo Jauregi Unanue, Ehsan Zare Borzeshi, Nazani Esmaili and Massimo Piccardi, “Low-Resource Natural Language Processing”, Research Bites, Spanish Researchers in Australia Pacific (SRAP), Sydney, Australia, 20th of June, 2019.

– Iñigo Jauregi Unanue, Ehsan Zare Borzeshi, Nazanin Esmaili and Massimo Piccardi, “Low-Resource Neural Machine Translation”, Weekly Seminar of the RoZetta Institute, Sydney, Australia, 13th of September, 2019.

1.3 Thesis Chapters

The thesis has been organized over eight chapters that are summarized as follows:

Chapter 1 The present chapter. The research topic and the contributions are introduced.

Chapter 2 This chapter contains the literature review which discusses traditional and more recent neural state-of-the-art models proposed in NER and MT. The chapter then presents the main research work conducted to deal with the overfitting problem on small datasets.

Chapter 3 In this chapter we describe our work on health-domain NER. We have focused on two particular tasks: drug name recognition and clinical concept extraction. Health-domain word embeddings have been pre-trained using a publicly available medical unsupervised dataset. Those embeddings are used as input features for a state-of-the-art NER model (i.e. BiLSTM-CRF). The experiments have shown that these embeddings improve the test accuracy and outperform previously proposed models over challenging benchmarks. The work presented in this chapter has been published in the Journal of Biomedical Informatics [Jauregi Unanue *et al.*, 2017].

Chapter 4 In this chapter a systematic comparison of three contemporary MT models is presented: an SMT model trained by ourselves, an NMT model trained by ourselves and an off-the-shelf commercial NMT system. We have tested these models over the English-

Basque low-resource language pair, using the limited annotated data that is publicly available. The experiments show that traditional SMT models can outperform state-of-the-art NMT systems when supervised data are scarce. The work in this chapter has been published in the Language Resources and Evaluation Conference 2018 [Jauregi Unanue *et al.*, 2018a].

Chapter 5 This chapter presents a novel regularization technique for NMT models. It explains how NMT models suffer from overfitting when trained with maximum likelihood estimation (MLE). The regularizer forces the model to jointly learn to predict the next word in the translation (categorical value) and its word embedding (continuous value). Additionally, we also propose the use of a second regularizer where the model learns to predict a sentence embedding. The experiments show that the use of this regularizer can consistently improve state-of-the-art NMT baselines over three low-resource language pairs. However, in a high-resource language pair, we observe that there is no need to use this regularizer. Further experiments have demonstrated that the regularizers work well because they help the model to learn a better clustered output space, hence, facilitating the target word classification. The work in this chapter has been published in the North American Chapter of the Association for Computational Lingusitics (NAACL) 2019 [Jauregi Unanue *et al.*, 2019] and a journal extension has been submitted to the IEEE Transactions on Neural Networks and Learning Systems (T-NNLS) and is currently under review.

Chapter 6 This chapter presents a novel way to train document-level NMT models combining reinforcement-style training with document discourse evaluation metrics. The approach presented in this chapter aims to explicitly train the model to improve document discourse using existing discourse rewards in the objective function. The experimental results showed that the proposed approach can improve discourse properties of the translated documents such as the lexical cohesion or coherence. At times, other accuracy based evaluation metrics such as BLEU are also improved using this training method. The work in this chapter has been submitted to an international NLP conference (ERA Rank A) in collaboration with researchers Dr. Nazanin Esmaili (University of Technology Sydney) and Prof. Gholamreza Haffari (Monash University), and my supervisor, Prof. Massimo Piccardi.

Chapter 7 This chapter presents the work done in automatic post-editing (APE). APE is a subfield of machine learning, where a fine-tuned APE model is trained to correct the errors of a general MT model. Our work has proposed a novel shared attention mechanism that contributes to improving the translation accuracy and the interpretabilty of the model. The work in this chapter has been published in the 2^{nd} Workshop on Neural Machine Translation and Generation (WNMT 2018), held in conjunction with ACL 2018. [Jauregi Unanue *et al.*, 2018b].

Chapter 8 This chapters concludes the dissertations and discusses possible future research work in the area of low-resource NER and MT.

Chapter 2

Literature Review

The literature review presented in this chapter is organized in the following way. First, Sections 2.1 and 2.2 frame the previous work in NER and MT, respectively. These two sections focus mainly on the description of the tasks, the conventional machine learning models proposed before the emergence of deep learning, and the standard way of evaluating the models. Then, Section 2.3 introduces the related deep learning models and training techniques. NER and MT share many neural architectures, and it seemed more convenient to describe them jointly in a separate section. Finally, Section 2.4 discusses the main related work on low-resource NER and MT.

2.1 Named-Entity Recognition

From a machine learning perspective NER is a sequential classification problem. Given an input sentence $x = \{x_1, x_2, \ldots, x_n\}$ the system needs to predict a sequence of labels $y = \{y_1, y_2, ..., y_n\}$, where each x_i represents a word in the text and y_i its corresponding *named entity*. Typical predefined named entities are: person names, organizations, locations, currency, URLs and time. Fig. 2.1 shows an example of a sentence with some predefined entities. In this example, 'John' is identified as a *person name* ('PER') and 'University of Technology Sydney' is identified as a *organization* ('ORG'). Note that a named entity can contain multiple words. The IOB2 tagging format [Sang and Veenstra, 1999] introduces additional markers in order to accurately define the boundaries of multiple token named entities. 'B-' (*beginning*) indicates that the word is the first token of a named entity, 'I-'

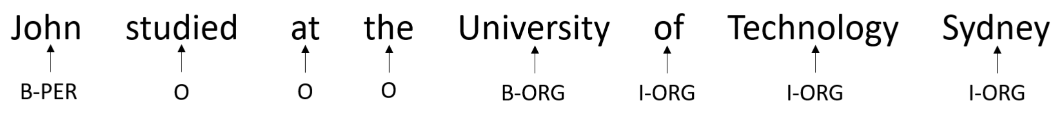


Figure 2.1: Example of a sentence annotated with named entities according to the IOB2 tagging format.

(*inside*) indicates that the token is still part of the same entity and 'O' (*outside*) indicates that he word does not belong to any predefined class.

Nevertheless, NER is not limited to only identifying those types of named entities. Depending on the domain and the application, NER models can be trained to find other classes. An interesting domain with many NER applications is health. Chapter 3 of this thesis, for example, proposes some improvements to two typical NER tasks in the health domain. These include drug name recognition [Chalapathy *et al.*, 2016b], clinical concept extraction [Chalapathy *et al.*, 2016a], de-identification of patients on electronic health records [Dernoncourt *et al.*, 2017], finding entities in the biomedical domain [Gridach, 2017] or identifying mentions of adverse reactions of drugs in social media [Cocos *et al.*, 2017]. The main problem of health-domain NER research is usually the limited availability of annotated data.

2.1.1 Traditional NER systems

Initially, NER relied completely on hand-crafted rules in order to extract these classes. An example of that is the first known NER system proposed by [Rau, 1991]. The NER proposed in this paper could extract company names from text using different lexical and syntactic rules such as word capitalization, acronym matching, checking co-occurences of prepositions with nouns, and so on. However, researchers soon realised that these approaches could not scale, because hand-crafted rules are very domain-dependent and do not generalize well to different text domains and different languages.

As a result, researchers started to move on towards language independent statistical models [Nadeau and Sekine, 2007]. Since the mid-90s, several conferences and workshops on NER have been organized, which has led to the release of many annotated NER corpora [Grishman and Sundheim, 1996; Chinchor *et al.*, 1998; Sang and De Meulder, 2003; Doddington *et al.*, 2004; Santos *et al.*, 2006; Pradhan *et al.*, 2013]. These datasets can differ in terms of the target entity classes (e.g company names, people names, drug names, mentions of clinical events...), the domain of the documents/sentences (e.g. medical, news, finance, fiction...) and the size of annotated data. The main challenge remains on finding adequate training data for your target NER tasks. In any case, researchers have used these datasets to train machine learning models that are able to learn disambiguation rules from the data. Traditional machine learning models have usually followed a two-step supervised learning approach. First, a *feature engineering* step converts the input tokens into numerical vectors. These features are designed by humans in order to embed information that is helpful for the task. Second, a sequential classifier is trained to learn the best parameters that maximize the likelihood of the training data (features and annotated labels). The classifier is considered *generative* if it is trained to maximize the joint probability of the training data:

$$\theta^* = \arg\max_{\theta} p(x, y|\theta) \tag{2.1}$$

where θ are the parameters of the model. Instead, the classifier is considered *discriminative* if it is trained to maximize the conditional probability of the output sequence given the input words:

$$\theta^* = \arg\max_{\theta} p(y|x, \theta) \tag{2.2}$$

During inference, the model selects the output sequence that maximizes the trained probability (*generative*(2.3) or *discriminative*(2.4))

$$y^* = \arg\max_{y} p(x, y|\theta^*) \tag{2.3}$$

$$y^* = \arg\max_{y} p(y|x, \theta^*) \tag{2.4}$$

Researchers have proposed both generative and discriminative classifiers for NER. Typical generative approaches are based on Hidden Markov Models (HMMs) [Bikel *et al.*, 1997; Morwal *et al.*, 2012], while discriminative approaches rely on Maximum Entropy models (ME) [Borthwick *et al.*, 1998], Conditional Random Fields (CRFs) [McCallum and Li, 2003] and Support Vector Machines (SVMs) [Asahara and Matsumoto, 2003].

However, in recent years, if there are sufficient supervised data to train the models, the aforementioned "traditional" machine learning models have been outperformed by deep learning models. Sequential deep learning models are usually discriminative classifiers that make use of advanced neural architectures such as recurrent neural networks (RNNs) or *transformers* in order to fit the training data. Section 2.3 will describe these models in more detail and how they have been used in NER.

2.1.2 Evaluation in NER

Automatic quantitative evaluation is a fundamental aspect of empirical research that permits a systematic comparison of different models. The standard accuracy, which computes the ratio of the number of correct predictions and the number of total predictions, is a typical evaluation metric in many NLP systems. However, the accuracy is a poor metric for imbalanced datasets such as NER's. In NER tasks, most of the words in the text do not belong to any predefined named-entity class and are labelled as 'O'. Let's assume, for instance, that 80% of the words belong to class 'O' and the rest of the words belong to some named entity: then, a dummy classifier that does not predict any named entity and which output is always 'O' will get an accuracy of 80%, which seems misrepresentative.

For that reason, the NER community has decided to generally use the *F1-score* (2.5). The *F1-score* is the equally weighted harmonic mean of the *precision* and *recall*. This metric better captures the quality of a NER system. However, different ways of computing the precision and recall have been proposed such as the CoNLL-F1, ACE, MUC-7 and SemEval. Here we address the most common one used by researchers: CoNLL-F1.

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.5}$$

2.1.2.1 CoNLL-F1

This version of the F1 was presented at the CoNLL-2003 named-entity recognition shared task [Sang and De Meulder, 2003] and it is a very common and widespread metric for the evaluation of NER systems. CoNLL-F1 is a strict version of the normal F1 due to the fact that it only considers as true positives named entities that are an exact match of the ground truth (i.e. all tokens). For example, if a named entity has 3 tokens and the classifier has only missed one of them, this evaluation metric will consider that the prediction is fully incorrect, even if two of the tokens have been correctly classified. The whole named entity chunk needs to be correctly classified. Thus, the precision and recall are computed in the following way:

$$precision = \frac{\#correct\ chunks}{\#guessed\ chunks} \qquad recall = \frac{\#correct\ chunks}{\#true\ chunks} \tag{2.6}$$

2.2 Machine Translation

Machine translation, too, can be considered a sequential classification task; yet, it is very different from NER. In machine translation, a *source* sentence $x = \{x_1, x_2, \ldots, x_n\}$ with n tokens has to be translated into a sentence in the *target* language $y = \{y_1, y_2, \ldots, y_m\}$ with m tokens. Figure 2.2 shows a translation of a sentence from Basque to English. There are two key differences that make MT a more challenging task compared to NER:

1. Unlike in NER, the target sequence does not need to contain the same number of tokens as the source ($\exists\ n \neq m$). This means that the machine translator has to learn to make predictions of sentences with variable length. In Figure 2.2 the source sentence has 4 words, while the target sentence has 6 words.

2. The output space in MT is very large. In NER, the classifier needs to choose between a small set of predefined named entities (e.g. 10$\sim$20), while in MT each word in the vocabulary of the target language becomes an output class. For example, the Oxford English Dictionary contains full entries for 171,476 words. In practice the vocabulary is usually trimmed to the most frequent $\sim$50,000 words in the training data, which is still much larger than in NER.

2.2.1 Traditional MT systems

MT is probably one of the oldest tasks in NLP as it has been around since the 1950s. The early MT models relied completely on handcrafted-rules and heuristics derived from well grounded linguistics theories. These include rule-based models (RBMT)[Appelo, 1986], constraint-based models (CBMT)[Arnold and Sadler, 1992], knowledge-based models (KBMT)[Carbonell *et al.*, 1978], lexical-based models (LBMT)[Abeillé *et al.*, 1990], principle-based models (PBMT)[Dorr, 1991] ans shake and bake models (S&BMT)[Beaven, 1992].

However, the limitations of rules soon became manifest, as they depend too much on the translation domain and language, and consequently have a very little capacity of generalization. Thus, similar to NER, researches started to develop language independent, statistical machine translation (SMT) models [Brown *et al.*, 1990; Doi and Muraki, 1992; Nomiyama, 1992]. SMT models use a Bayesian approach to model the probability of a target sentence given the source sentence:

$$p(y|x) \sim p(x|y) \times p(y) \tag{2.7}$$

where $p(y)$ is a language model that evaluates the correctness of translation y in the target language, and $p(x|y)$ models the likelihood of x being the source sentence given the translation y. The probabilities of this equation are learned from the translation corpus.

Among the different SMT paradigms that have been proposed in MT research, the phrase-based statistical machine translation (PB-SMT) has become the dominant model [Koehn *et al.*, 2003]. In PB-SMT, the source sentence is first sampled into a sequence of I phrases $\overline{x}^I$ with uniform distribution. Each phrase $\overline{x}_i$ can be a single token or multiple tokens in the source sentence. Then, all the phrases are translated into the target language with a conditional phrase translation probability model $\phi(\overline{x}_i|\overline{y}_i)$. The translated phrases are re-ordered in the target language using a distortion probability distribution $d(a_i - b_{i-1})$

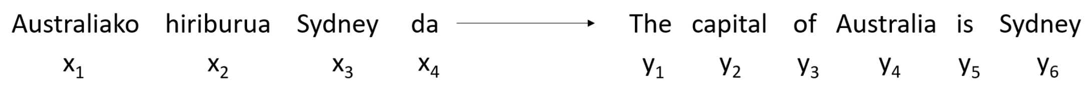


Figure 2.2: Example of a translation from Basque to English.

where a_i and b_{i-1} denote the start and end positions of the source phrases translated into the i^{th} and $(i-1)^{th}$ target phrases respectively. In addition, the length of the translation is also calibrated by introducing a ω factor. As a result, the probability of the segmented translation $p(\overline{y}^I|\overline{x}^I)$ is modeled as:

$$p(\overline{y}^I|\overline{x}^I) \sim p(\overline{x}^I|\overline{y}^I) \times p(\overline{y}^I) \times \omega^{length(\overline{y}^I)} \tag{2.8}$$

where $p(\overline{x}^I|\overline{y}^I)$ is decomposed into:

$$p(\overline{x}^I|\overline{y}^I) = \prod_{i=0}^{I} \phi(\overline{x}_i|\overline{y}_i) d(a_i - b_{i-1}) \tag{2.9}$$

At inference, the model decodes the source sentence from left to right with the beam-search algorithm. It starts with an empty hypothesis, translating each phrase in the source sentence one by one. At the end, the hypothesis that maximizes the probability is chosen as the translation:

$$y^* = \arg\max_y p(y|x) = \arg\max_{\overline{y}^I} p(\overline{x}^I|\overline{y}^I) \times p(\overline{y}^I) \times \omega^{length(\overline{y}^I)} \tag{2.10}$$

The PB-SMT has been the *de-facto* model in MT for many years thanks to the release of the Moses open source translation toolkit [Koehn *et al.*, 2007]. This framework has facilitated the building and training of these models, including training of phrase alignment models such as Giza++ [Och and Ney, 2003] and language models such as KenLM [Heafield, 2011].

Currently, the neural sequence-to-sequence models have outperformed PB-SMT when trained on large available corpora [Bojar *et al.*, 2016; Bojar *et al.*, 2017; Sutskever *et al.*, 2014], as it has happened in NER. However, when the amount of training data is limited, PB-SMT can still outperform deep learning models.

2.2.2 Evaluation in MT

The quality of a translation is evaluated in two directions: *adequacy* and *fluency*. The former measures the information rate that has been correctly translated in the target sentence, while the latter evaluates the grammatical correctness of the sentence. Yet the

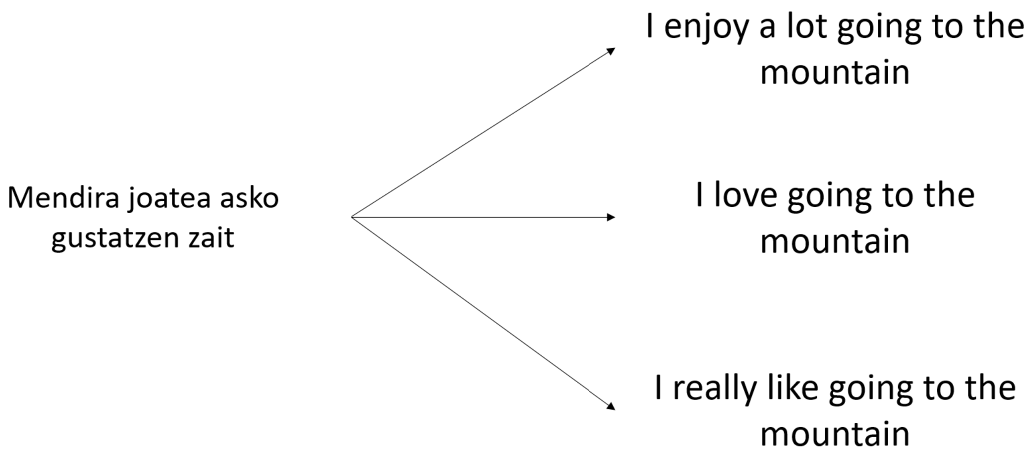

Mendira joatea asko gustatzen zait
I enjoy a lot going to the mountain
I love going to the mountain
I really like going to the mountain

Figure 2.3: A source sentence in Basque can be translated into multiple correct sentences in English.

automatic evaluation of these attributes over the sentences generated by the MT models is a challenging task and is still considered an open problem. The main challenge is that a sentence in a source language can have multiple translation that are equally correct. See, for instance, the example in Figure 2.3. The Basque sentence "*Mendira joatea asko gustatzen zait*" has three correct English translations, which are grammatically correct and preserve the meaning of the original sentence. However, it is difficult to find translation datasets that, for each source sentence, cover all the space of correct translations. Most of the available translation corpora contain one ground-truth reference per source sentence.

In this scenario, one obvious evaluation method is human evaluation. Multiple professional translators can score the predictions made by the MT systems. But this approach is expensive, time-consuming and often the inter-annotator agreement of the translators is low [Stymne and Ahrenberg, 2012]. As a result, the researchers in the field have been devoted to developing alternative automatic evaluation metrics that correlate well with human judgment and, among them, BLEU [Papineni *et al.*, 2002] has become de facto metric. The idea is to compute a weighted average of n-gram matches against the ground-truth reference translation. For that, the BLEU's authors have proposed a modified n-gram precision score:

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')} \tag{2.11}$$

that makes a clipped count of the total number of n-grams in the candidate predictions of the MT system that overlap with the ground-truth reference ($Count_{clip}$ only counts the maximum number of times an n-gram appears in a single reference), divided by the total number of n-grams in the predictions. Then, a weighted average of the different n-gram precision scores (e.g. unigram, bigram, trigram) is performed, including a sentence brevity factor BP:

$$BLEU = BP \times exp(\sum_{n=1}^{N} w_n \log p_n) \tag{2.12}$$

$$BP = \begin{cases} 1 & if \quad c > r \\ \exp^{(1-r/c)} & if \quad c \leq r \end{cases} \tag{2.13}$$

where c represents the candidate sentence predicted by the MT model, r is the ground-truth reference, N is the maximum number of words in an n-gram and $w_n = 1/N$.

Another widely used MT evaluation metric is the Translation Edit Rate (TER) [Snover *et al.*, 2006], which measures the minimum number of editing that needs to be done to the prediction in order to exactly match the reference. Possible edits include substitution of single words, insertion of new words, deleting words or shifting words in the sequence. Thus, the TER metric is computed dividing the number of edits made over the hypothesis by the total number of words in the reference sentence:

$$TER = \frac{\#\ of\ edits}{\#\ of\ words\ in\ the\ reference} \tag{2.14}$$

The authors claim that TER has higher correlation with human than BLEU and other MT evaluation metrics such as METEOR [Banerjee and Lavie, 2005] or NIST [Doddington, 2002], which also follow a similar intuition.

However, the aforementioned n-gram matching based metrics have many limitations. For example, Callison-Burch et al. [2006] have claimed that the correlation between the BLEU score and the human-judgment has been often overestimated. On the one hand, they are not good at capturing distant phrasal dependencies and penalizing the swap of cause and effect clauses [Isozaki *et al.*, 2010] (e.g. *A because B* instead of *B because A*). On the other hand, these metrics can only recall n-grams that exactly occur in the reference translations and are unable to account for the meaning-preserving lexical and compositional diversity of translations such as synonyms (METEOR excepted) and paraphrases [Zhang *et al.*, 2020]. As a consequence, in recent years there has been a trend to propose alternative evaluation metrics in workshops and conferences (e.g. WMT [Bojar *et al.*, 2016; Bojar *et al.*, 2017; Bojar *et al.*, 2018; Barrault *et al.*, 2019]).

Recently, there has been a proposal to produce word embedding based evaluation metrics (see Section 2.3.1 for more information on word embeddings). Zhang et al. [2020] have proposed a new metric named F_{BERT}, which uses contextual word embeddings learned from a pre-trained BERT model [Devlin *et al.*, 2019] to evaluate the translation quality. Similar to the famous $F1$-score, F_{BERT} is the harmonic-mean of the precision (P_{BERT}) and the recall (R_{BERT}):

$$R_{BERT} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^T \hat{\mathbf{x}}_j \tag{2.15}$$

$$P_{BERT} = \frac{1}{|x|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^T \hat{\mathbf{x}}_j \tag{2.16}$$

$$F_{BERT} = 2\frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}} \tag{2.17}$$

where x_i is the i-th word in the tokenized reference sentence, $\mathbf{x}_i$ its contextual word embedding, $\hat{x}_j$ is the j-th word in the tokenized predicted sentence and $\mathbf{x}_j$ its contextual word embedding. According to Zhang et al. [2020], their proposed BERT-based evaluation metric shows stronger system-level and segment-level correlations with human judgment than other common metrics such as BLEU.

2.3 Deep Learning for NER and MT

Nowadays, the state-of-the-art machine learning models that have achieved the highest performance on most of the publicly available NER and MT benchmarks are based on deep neural models. The revolution of deep learning has contributed to very accurate systems and has brought NLP to the greater attention of the media, companies, investors and academia. In this section, the most common deep learning architectures for NER and

MT are described.

2.3.1 Word Embeddings

One of the great advantages of neural networks with respect to traditional machine learning systems is their ability to learn patterns from random input representations. This avoids time-consuming feature engineering as the neural network can act as a very good feature extractor. However, the performance of the network can vary substantially depending on the random seed. An unfortunate initialization of the features can make the training less effective.

In many NLP tasks, this problem can be alleviated by initializing words (input features) with meaningful pre-trained word embeddings [Li and Yang, 2018]. Word embeddings are high-dimensional (but much lower than one-hot encodings), dense vector representations that aim to capture the distributional and semantic properties of words in a particular language. The underlying assumption is that words with similar meaning appear in similar contexts. Following that assumption, unsupervised word embedding algorithms make use of word coocurrences in text in order to come up with good vector representations. As a result, embeddings of similar words are closer in the vector space. Additionally, these vectors tend to hold nice properties such as "$king - queen = man - woman$". Overall, word embeddings provide a better feature initialization for the neural network and facilitate the training.

In this section some of the most common unsupervised word embedding representations are described.

2.3.1.1 Word2vec

Word2vec [Mikolov *et al.*, 2013a] is an unsupervised word embedding algorithm based on the simplified architecture of a feedforward neural network language model (NNLM) in order to improve training efficiency. Word2vec can learn high-quality continuous vector representations by modeling the log-probabilities of word coocurrences in a fixed context window. The authors have proposed two variants of the algorithm: **skip-gram** and **continuous bag-of-words** (see Figure 2.4).

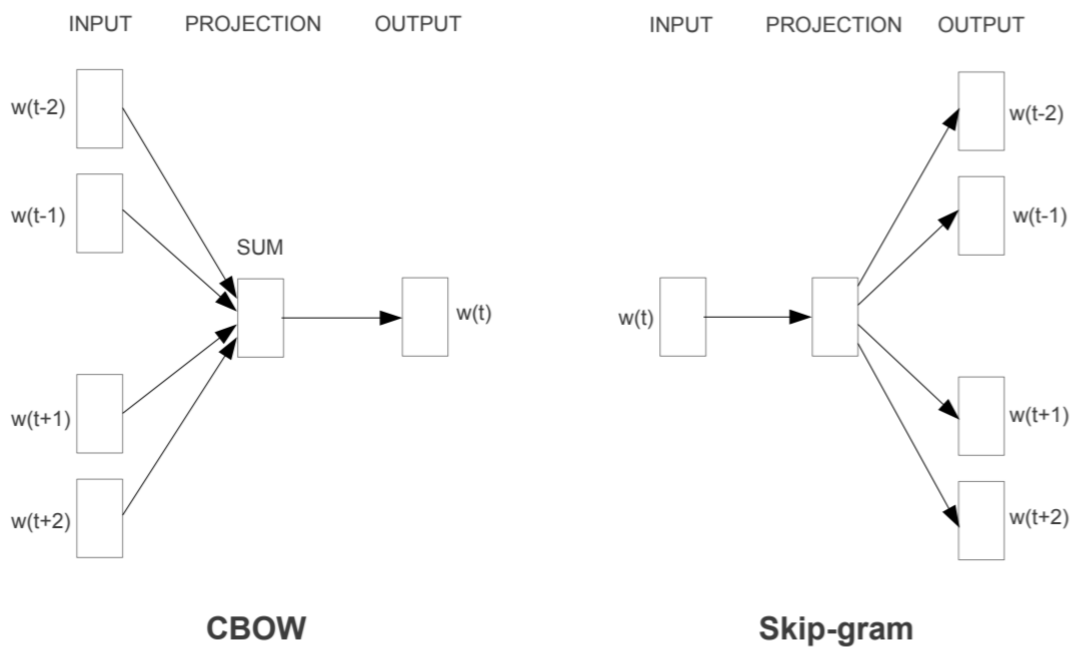


Figure 2.4: A figure from [Mikolov *et al.*, 2013a] which depicts the two variants of word2vec: CBOW (left) and Skip-gram (right).

Skip-gram has been described in detail in [Mikolov *et al.*, 2013b]. The model is trained to find vector representations that are good at predicting surrounding words in a sentence (or document):

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\ne 0}\log p(w_{t+j}|w_t) \tag{2.18}$$

where $w_1, \dots, w_T$ are words in the training batch and c is the context window size. During training, the model tries to maximize the average log-probability of a context word w_{t+j} given a center word w_t. The probability $p(w_O|w_I)$ is defined as a softmax function:

$$p(w_O|w_I) = \frac{exp({\mathbf{v}'_{w_O}}^T \mathbf{v}_{w_I})}{\sum_{w=1}^{W} exp({\mathbf{v}'_w}^T \mathbf{v}_{w_I})} \tag{2.19}$$

where $\mathbf{v}_{w_I}$ is the "input" vector representation and $\mathbf{v}'_{w_O}$ is the "output" vector representation and W is the number of words in the vocabulary. However, the computation of the denominator in this probability is intractable because it requires to compute the exponential for all the words in the vocabulary, which is very large (50,000 $\sim$ 100,000 words). The authors have proposed two efficient alternatives to compute an approximation of (2.19), the hierarchical softmax [Morin and Bengio, 2005] and Noise Constrastive Estimation (NCE) [Gutmann and Hyvärinen, 2012; Mnih and Teh, 2012].

CBOW is very similar to Skip-gram, but instead of predicting the context words given the centre word, the algorithms learns representations that help to predict the centre word given the context words:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c,j\neq 0}\log p(w_t|w_{t+j}) \tag{2.20}$$

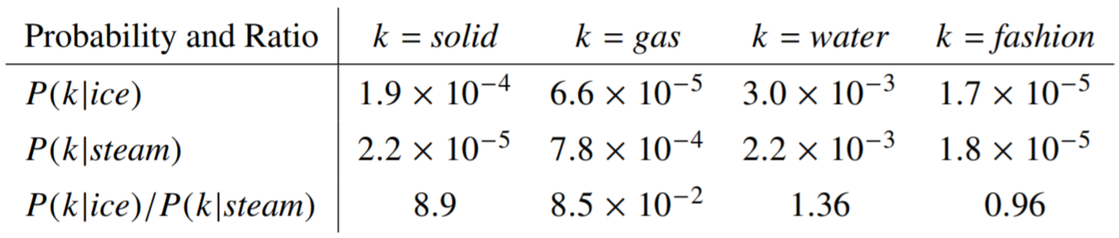
2.3.1.2 GloVe

GloVe, proposed by [Pennington *et al.*, 2014], is another unsupervised word embedding algorithm, which also looks at the co-occurrences of words in text in order to come up with good representations. The authors of GloVe point out that CBOW and Skip-gram make poor use of the statistics of the unsupervised corpus because they operate in local context windows. Therefore, they propose a global log-bilinear regression model that is trained on global word to word co-occurrence counts. Once the co-occurrence matrix for the corpus is calculated, the probability of a word j appearing in the context of word i is computed:

$$p_{ij} = p(j|i) = \frac{X_{ij}}{X_i} \tag{2.21}$$

where X_{ij} is the number of times those words co-occur and X_i is the number of times i appears together with any word in the context ($X_i = \sum_k X_{ik}$).

The table in Figure 2.5 shows a few examples of the co-occurrence probabilities for *target* words *steam* and *ice*, computed from a 6 billion token corpus. The table shows that



Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Figure 2.5: Some probabilities derived from the co-occurence matrix of a 6 billion token corpus [Pennington *et al.*, 2014].

relative probabilities $p(k|ice)/p(k|steam)$ are better than raw probabilities at discriminating relevant words (*steam* and *gas*, *ice* and *solid*) from irrelevant words (*ice* and *fashion*). Consequently, they start building a learning model that is based on ratios of probabilities:

$$F(\mathbf{w}_i, \mathbf{w}_j, \widetilde{\mathbf{w}}_k) = \frac{p_{ik}}{p_{jk}} \tag{2.22}$$

where $\mathbf{w} \in \mathbb{R}^d$ and $\widetilde{\mathbf{w}} \in \mathbb{R}^d$ are word vectors and context word vectors respectively. After several derivations (for more information the reader can check the original paper), they come up with a weighted least square objective function:

$$J = \sum_{i,j=1}^{|V|} f(X_{ij})(\mathbf{w}_i^T \tilde{\mathbf{w}}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \tag{2.23}$$

where $|V|$ is the size of the vocabulary, b is the bias term and $f(X_{ij})$ is the weighting function.

2.3.1.3 FastText

[Bojanowski *et al.*, 2017] have proposed an extension of the word2vec's Skip-gram model. Their model is able to learn representations for character n-grams or subword units of words, modeling the internal structure of words, which is very important for morphologically rich languages such as Turkish, Finnish or Basque.

FastText aims to learn vector representations that can predict context words given a central word. Similar to the Skip-gram model, it models this conditional probability:

$$p(w_c|p(w_t) = \frac{e^{s(w_t,w_c)}}{\sum_{j=1}^{W} e^{s(w_t,j)}} \tag{2.24}$$

where W is the vocabulary size, w_t is the center word and w_c is the context word. However, differently from word2vec (2.19), it uses a scoring function $s(w_t, w_c)$ instead of directly computing the inner product of the vectors. The scoring function is defined as:

$$s(w, c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^T \mathbf{v}_c \tag{2.25}$$

where $\mathcal{G}_w \subset \{1, \ldots, G\}$ is the subset of n-grams that appear in the word w and are

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder $\{\ldots\}$	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson $\{\ldots\}$	$\{\ldots\}$ they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Figure 2.6: An example from [Peters *et al.*, 2018]. They show the nearest neighbour words of the word *play* for GloVe, and the nearest neighbour sentences that contain the word *play* for the contextual embeddings.

included in a dictionary of n-grams G, $\mathbf{z}_g$ is the vector representation of the n-gram g and $\mathbf{v}_c$ is the vector representation of the word in the context c.

In the original paper, the authors carry out several experiments over 9 languages with different morphologies that demonstrate the benefits of using this model.

2.3.1.4 Contextual word embeddings

Recently, there has been a remarkable improvement regarding word embeddings. The latest models are using pre-trained language models in order to infer *contextual word embeddings*. This type of embeddings addresses a limitation of traditional ones: the variation of the word's meaning across different linguistics contexts. See the example in Figure 2.6. When using GloVe embeddings, the word *play* is shown to be closest to words related to sports such as *game*, *players* or *football*. Conversely, when computing the nearest neighbour of the contextual embeddings, we observe that depending on the context, word *play* can be closer to words in the topic of theatre.

Most of the state-of-the-art natural language understanding (NLU) systems use pre-trained language models to compute contextual embeddings and use them in their models. The best known pre-training language model algorithms are ELMo [Peters *et al.*, 2018], GPT [Radford *et al.*, 2018], BERT [Devlin *et al.*, 2019] and XLNet [Yang *et al.*, 2019]. A more detailed description of pre-trained language models is given in Section 2.4.6.

2.3.2 Sentence embeddings

In parallel to word embeddings, researchers have been interested in obtaining meaningful representations for larger sequences of words such as sentences, paragraphs or documents. These vectors are usually known as sentence embeddings and they can be very useful for many NLP applications such as unsupervised text clustering [Huang, 2008], automatic parallel translation corpus mining [Artetxe and Schwenk, 2019a] and many NLU tasks [Wang *et al.*, 2019].

The simplest and most intuitive way of obtaining sentence embeddings is to average the word embeddings in the sentence or document in order to obtain a fixed-dimensional vector. However, when the sentence is long, this approach does not work very well. Other more sophisticated algorithms include Skip-Thought vectors [Kiros *et al.*, 2015], which use the continuity of text from books to train an encoder-decoder model that tries to reconstruct the surrounding sentences of an encoded passage; InferSent [Conneau *et al.*, 2017], where sentence embeddings are learned in a supervised manner over the Stanford Natural language Inference dataset; Doc2vec [Le and Mikolov, 2014], an unsupervised embedding algorithm that is trained to predict words in the same sentence/document; SentenceBERT [Reimers and Gurevych, 2019], sentence embeddings produced by pooling the contextual word embeddings predicted by a pretrained LM such as BERT; massively multilingual sentence embeddings [Artetxe and Schwenk, 2019b], a sentence embedder formed by a single BiLSTM encoder that shares a vocabulary with 93 languages, which enables the model to create cross-lingual sentence embeddings; or the universal sentence encoder (USE) [Cer *et al.*, 2018], a transformer-based sentence embedder trained in a multi-task setting.

Among them, USE has been one of the most popular until recently thanks to its remarkable performance in NLU tasks and ease of use, with publicly available pre-trained models shared and ready to use with the well-known *tensorflow-hub* package [1]. The model can use two different architectures, one based on the *Transformer* [Vaswani *et al.*, 2017] and the other on *Deep Averaging Networks (DAN)* [Iyyer *et al.*, 2015]. In both cases, the sentence encoder is trained with multi-task learning, aiming to obtain very general vector representations that can perform well in a variety of tasks such as sentiment clas-

[1]USE model: https://tfhub.dev/google/universal-sentence-encoder/3

sification [Pang and Lee, 2005; Hu and Liu, 2004], document classification [Li and Roth, 2002], sentence similarity [Cer *et al.*, 2017] and natural language inference [Bowman *et al.*, 2015].

2.3.3 Recurrent Neural Networks

Tweets, paragraphs, documents or novels are all sequences of words, and words are sequences of characters at their turn. Words by themselves have meaning, but the meaning of a sentence obviously depends on the order of its words and not just their values. For example, the meaning of the two sentences "*I live in Sydney which is in Australia*" and "*I live in Australia which is in Sydney*" is completely different. Thus, models that are able to capture the positional inter-dependencies between the words in a sequence are needed.

RNNs are a type of neural networks that are effective for processing sequential data. They are an extension of feedforward networks in which cyclical or recurrent connections are allowed [Graves, 2012]. These recurrent connections create an internal "memory", which allows processing a sequence of vectors $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ as input and produce another sequence $(\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n)$ as output that contains some extent of sequential information about every vector in the previous elements in the input sequence. The first RNN is credited to [Hopfield, 1982] and is known as the Hopfield Network. However, RNNs became more popular in the 1990s when many other variants were proposed [Elman, 1990; Jordan, 1990; Lang *et al.*, 1990]. In the following sections different RNN models are described.

2.3.3.1 Vanilla RNNs

[Elman, 1990] and [Jordan, 1990] proposed two very similar, simple variants of the RNNs in the early 1990s, which are commonly known as Vanilla RNNs. This section describes these two models. To keep the description simple, consider a single layer RNN. Let's assume that $\mathbf{X} : \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ is the input sequence and that $\mathbf{Y} : \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$ is the sequence that needs to be predicted. The forward pass of the Elman network follows Equations (2.26). As can be seen, the model computes its internal hidden state $(\mathbf{h}_t)$ at time t, given the current element in the input sequence $(\mathbf{x}_t)$ and the previous hidden state

($\mathbf{h}_{t-1}$), together with some additional learnable parameters $\mathbf{W}_h$, $\mathbf{U}_h$ and $\mathbf{b}_h$. Note that $\mathbf{h}_{t-1}$ is the recurrent connection that retains the information from previous steps in the sequence ("memory"). Then, the output $\mathbf{y}_t$ is computed given $\mathbf{h}_t$ and the learnable parameters $\mathbf{W}_y$ and $\mathbf{b}_y$. σ_h and σ_y are non-linear functions such as *sigmoid* or *tanh*.

$$\begin{aligned} \mathbf{h}_t &= \sigma_h(\mathbf{W}_h\mathbf{x}_t + \mathbf{U}_h\mathbf{h}_{t-1} + \mathbf{b}_h) \\ \mathbf{y}_t &= \sigma_y(\mathbf{W}_y\mathbf{h}_t + \mathbf{b}_y) \end{aligned} \tag{2.26}$$

The Jordan network only differs from the Elman network in the recurrent connections. Instead of using the previous hidden state ($\mathbf{h}_{t-1}$), it uses the previous output ($\mathbf{y}_{t-1}$) (Equation 2.27).

$$\begin{aligned} \mathbf{h}_t &= \sigma_h(\mathbf{W}_h\mathbf{x}_t + \mathbf{U}_h\mathbf{y}_{t-1} + \mathbf{b}_h) \\ \mathbf{y}_t &= \sigma_y(\mathbf{W}_y\mathbf{h}_t + \mathbf{b}_y) \end{aligned} \tag{2.27}$$

Figure 2.7 shows a diagram of the Elman Network that may help to understand it better. On the left, it shows an RNN unit with the two inputs, the current element of the input sequence and the previous hidden state. Given those inputs, the system predicts the next hidden state and the output. On the right, we see the same RNN as a time-unfolded graph, where each node corresponds to a particular time step. It can be seen that $\mathbf{y}_t$ depends on all the previous elements in the input sequence $\mathbf{x}_1, \ldots, \mathbf{x}_t$. Thus, the network keeps all the previous states in "memory" to predict the next value. Note that Figure 2.7 corresponds to a sequential prediction example. For each input $\mathbf{x}_t$, an $\mathbf{y}_t$ is predicted. In other words, the input and the output have the same length. This is a good example of an NER system. For each word in the input, a named-entity class is predicted. However, that is not always the case. For example, in MT, the predicted sequence does not have to be of the same length of the input. In such cases, the sequence of hidden states ($\mathbf{h}_1, \ldots, \mathbf{h}_n$), needs to be converted into a fixed dimensional vector, which will later be used to predict the output sequence. A common way of encoding the input sequence into a fixed dimensional vector is to only keep the last hidden state, which in theory will retain the information of the whole sequence. Other more sophisticated approaches learn attention weights to compute a weighted sum of the hidden states [Bahdanau *et al.*, 2015; Luong *et al.*, 2015]. The hidden states with higher weights will have a greater influence

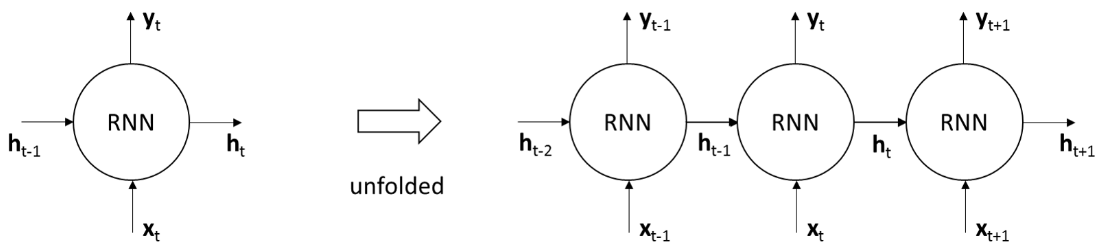


Figure 2.7: The Elman network. (*left*) An RNN unit. (*right*) The same RNN unit but time-unfolded.

in the prediction.

RNNs are trained in the same way to the feedforward networks with back-propagation [Rumelhart *et al.*, 1986]. In order to compute the gradient in RNNs, the generalized back-propagation algorithm needs to be applied to the unfolded graph shown in Figure 2.7. For a complete example of the computation of the gradient, please refer [Goodfellow *et al.*, 2016]. Nowadays, however, there is software available that allows computing the gradient automatically [Bergstra *et al.*, 2010; Collobert *et al.*, 2002; Abadi *et al.*, 2016]. This software usually include all the necessary tools to successfully train our RNNs, which is very beneficial for the research in this field.

To conclude, it is important to address the drawbacks of Vanilla RNNs. In theory, the internal memory of the network should help correctly predicting the output. Nevertheless, in practice they often fail to learn long-term dependencies in the sequences as they tend to be biased by the most recent vectors. The main problems are the exploding and vanishing gradients. The exploding gradient happens when large error gradients accumulate during back-propagation. This leads to big parameter updates, which can make the learning unstable [Goodfellow *et al.*, 2016]. A solution for the exploding gradient problem is gradient clipping [Pascanu *et al.*, 2013]. However, gradient clipping does not solve the vanishing gradient problem. The vanishing gradient happens when the gradients quickly tend to zero as the sequence increases. Consequently, short-term dependencies dominate the computation of gradient and training becomes more difficult [Bengio *et al.*, 1994].

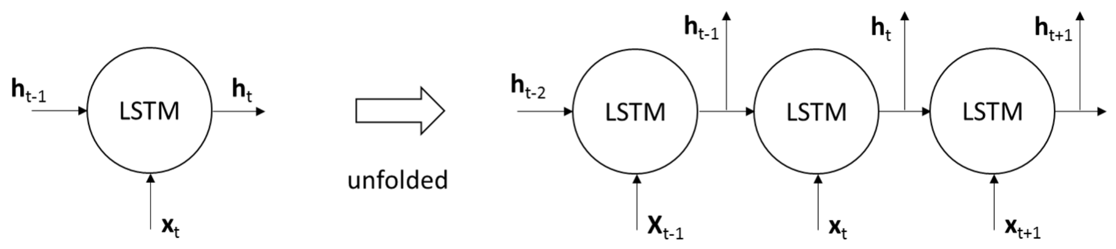


Figure 2.8: The LSTM network. (*left*) A LSTM unit. (*right*) The same LSTM unit but time-unfolded.

2.3.3.2 LSTM

In order to alleviate the vanishing and exploding gradient problems in vanilla RNNs, researchers have proposed gated RNNs such as the long short-term memory network (LSTM) [Hochreiter and Schmidhuber, 1997]. LSTMs follow the same cyclic structure with a repeating recurrent module (Fig. 2.8), but with a more complex internal architecture than a normal RNN as they contain some extra gates and cells that interact in a very specific way. The usual forward pass of the recurrent unit follows Equations 2.28. Note that in these equations we do not calculate the output vectors $\mathbf{y}_t$. Nevertheless, it is straightforward to apply a multi layer perceptron (MLP) and a soft-max function to each $\mathbf{h}_t$ to get the output values as it was done in Equations 2.26 and 2.27.

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{h}_t &= \mathbf{o}_t \circ \tanh(\mathbf{c}_t)
\end{aligned} \tag{2.28}$$

In Equation 2.28, for each t-th element in the sequence, **h** is the hidden state, **c** is the memory cell state, **f** is the forget gate, **i** is the input gate, **o** is the output gate, σ is the sigmoid function and $\circ$ is the Hadamard product. The memory cell is another separate recurrent connection, in addition to the hidden state, which stores the information of all the sequence. The forget gate controls the amount of information from the cell state that will flow through the network. The input gate permits to insert more information from the current state into the cell state. Finally, the output gate generates the new hidden state

($\mathbf{h}_t$), given the current element from the input sequence ($\mathbf{x}_t$), the previous hidden state ($\mathbf{h}_{t-1}$) and the current memory cell ($\mathbf{c}_t$). Figure 2.9 shows a more detailed picture of an LSTM unit.

The LSTM not only deals better than the RNN with the long-term dependencies, but also converges faster. Consequently, the LSTM is more efficient during training and less supervised data is required. The main drawback is that the number of learnable parameters is bigger, and thus, the amount of memory required to perform all the necessary computations is larger.

2.3.3.3 GRU

The GRU is another gated RNN recently proposed by [Cho *et al.*, 2014] to also solve the vanishing gradient problem. However, its internal mechanism differs considerably from the LSTM. The forward pass of the GRU (Equations 2.29) only has two gates: the update gate **z** and the reset gate **r**.

$$\begin{aligned}
\mathbf{z}_t &= \sigma(\mathbf{W}_z\mathbf{x}_t + \mathbf{U}_z\mathbf{h}_{t-1} + \mathbf{b}_z) \\
\mathbf{r}_t &= \sigma(\mathbf{W}_r\mathbf{x}_t + \mathbf{U}_r\mathbf{h}_{t-1} + \mathbf{b}_r) \\
\mathbf{h}_t &= (1 - \mathbf{z}_t) \circ \mathbf{h}_{t-1} + \mathbf{z}_t \circ \tanh(\mathbf{W}_h\mathbf{x}_t + \mathbf{U}_h(\mathbf{r}_t \circ \mathbf{h}_{t-1}) + \mathbf{b}_h)
\end{aligned} \tag{2.29}$$

The reset gate is equivalent to the forget gate in the LSTM as it controls the flow of previous states information. Similarly, the update gate is equivalent to the input gate of the LSTM as it manages the input of the new information into the hidden state. The separate

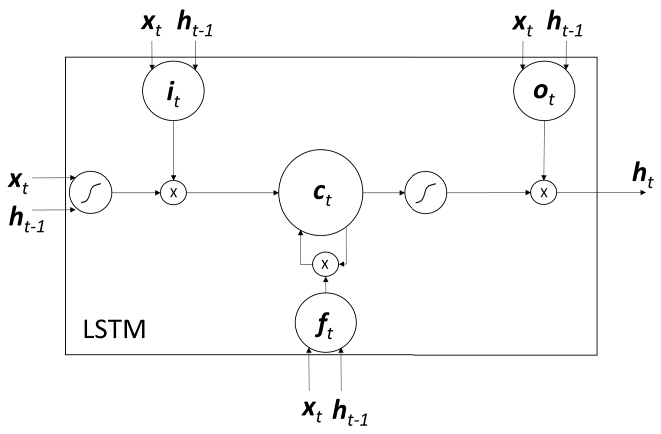


Figure 2.9: Detailed LSTM unit.

memory state cell (**c**) and the output gate (**o**) from Equations 2.28 are not present in the GRU. Figure 2.10 shows the detailed diagram of a GRU unit.

[Chung *et al.*, 2014] have compared these two gated RNNs on the tasks of polyphonic music and speech signal modelling. Their experiments have shown that the GRU has a comparable performance to the LSTM, while being computationally more efficient. Consequently, the number of research papers that use GRU networks has increased in the last few years.

2.3.3.4 Bidirectional RNNs

In the previous sections, several types of RNNs have been described. In all of them, the predicted hidden state ($\mathbf{h}_t$) depends on past information on the input sequence. But in none of these models the hidden state depends on future information of the sequence, which may often be useful, especially in NLP.

The Bidirectional RNN [Schuster and Paliwal, 1997] aims to learn to predict the value at each time step by looking at both the previous and the future tokens in the input sequence. The approach is very simple. Basically, two separate RNNs are learned. The first one learns the hidden representations of the sequence by reading the sequence left-to-right. The second RNN learns the hidden representation by reading the sequence backwards, right-to-left. Finally, the forward and the backward representations are concatenated to form a single representation (Equation 2.30). Figure 2.11 shows the typical structure of the Bidirectional RNN. Any kind of RNN can be used (e.g. vanilla, LSTM, GRU).

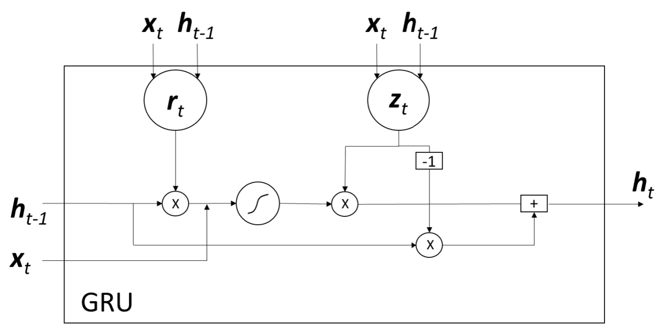


Figure 2.10: Detailed GRU unit.

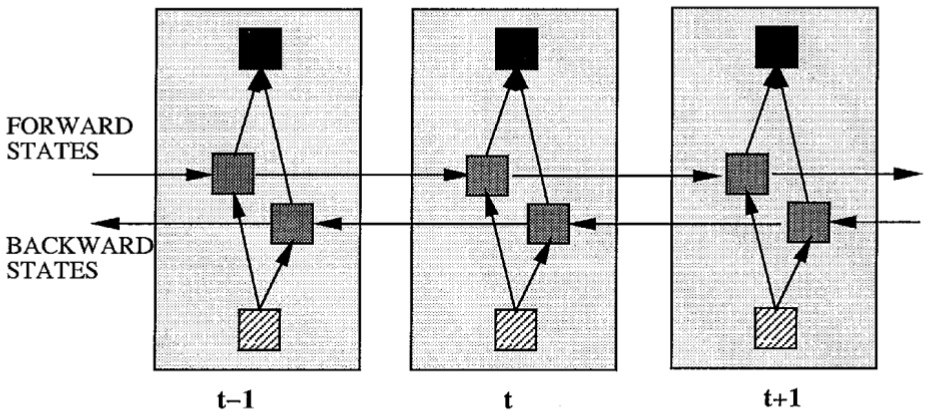


Figure 2.11: Bidirectional RNN [Schuster and Paliwal, 1997].

$$\begin{aligned}\mathbf{h}_r &= RNN_{right}(\mathbf{x}_t, \mathbf{h}_{t-1}) \\ \mathbf{h}_l &= RNN_{left}(\mathbf{x}_t, \mathbf{h}_{t+1}) \\ \mathbf{h} &= [\mathbf{h}_r; \mathbf{h}_l]\end{aligned} \tag{2.30}$$

2.3.4 Deep Sequential Classification

As mentioned in Section 2.1 and Section 2.2, NER and MT are both sequential classification tasks. In both cases, the classifier needs to learn how to predict a sequence of output labels given an input sentence. Yet there are big differences between the two. NER has to generate a label per token in the input, which results in an output sequence of the same length, and the number of output classes is quite small ($10 \sim 20$). Instead, in MT, the model needs to generate a sequence (the translation) of variable length which does not need to have the same length as the input and with a much larger ($30,000 \sim 80,000$) number of output classes. For these reasons, researchers have proposed different neural architectures for each task. This section covers two of them, which have been the workhorse architectures in their respective task: the *BiLSTM-CRF* and the *seq2seq* models. Additionally, describes a novel neural network proposed in the literature, namely the Transformer, which introduces some changes that improve limitations of RNNs.

2.3.4.1 BiLSTM-CRF

One of the most popular neural architecture for NER in recent years has been the BiLSTM-CRF [Lample *et al.*, 2016; Huang *et al.*, 2015]. The BiLSTM network is very good at encoding the input sequence into a sequence of latent variables $(\mathbf{h}_1, \dots, \mathbf{h}_n)$. A simple and effective strategy is to use these vectors together with a softmax layer to make a sequence of independent classification decisions. This has been quite successful in tasks such as POS tagging. However, NER is a task that is more dependent on the grammar and the context words in the sentence. For example, if a token is preceded by a 'B-PERSON' label, the chances of the next label being 'I-PERSON' are very high, and on the contrary, the chances of the next label being 'I-ORGANIZATION' are zero. Therefore, an output layer that allows structured prediction may be very helpful.

The CRF [Lafferty *et al.*, 2001] layer after the BiLSTM allows for that joint sequential prediction. The CRF uses the hidden vectors that are the output from the LSTM as measurements $\mathbf{P} \in \mathbb{R}^{n\times k}$ and a state transition matrix $\mathbf{A} \in \mathbb{R}^{(k+2)\times(k+2)}$, which can use the previous and future measurements to predict the current class. n is the number of tokens in the sequence and k is the number of output classes. Matrix $\mathbf{A}$ is a $k+2$ squared matrix because y_0 and y_{n+1} are the *start* and *end* labels. Given these two matrices, the score function of a given output sequence is computed by:

$$s(X, y) = \sum_{i=0}^{n} \mathbf{A}_{y_i, y_{i+1}} + \sum_{i=1}^{n} \mathbf{P}_{i, y_i} \tag{2.31}$$

Given the scoring function, being CRF a discriminative approach, it models the conditional probability $p(y|X)$ as:

$$p(y|X) = \frac{exp(s(X, y))}{Z(X)} \tag{2.32}$$

where $Z(X)$ is the cumulative sum of $exp(s(X, y))$ over all the possible y.

The parameters of the BiLSTM-CRF are typically jointly learned end-to-end over a training set, $(\mathbf{X}_{train}, \mathbf{Y}_{train}) = \{\mathbf{X}_i, \mathbf{Y}_i\}\ i = 1, \dots, N$, with conditional maximum likelihood (Equation 2.33).

$$\theta, A = \arg\max_{\theta, A} p(\mathbf{Y}_{train}|\mathbf{X}_{train}, \theta, \mathbf{A}) \tag{2.33}$$

where θ and A are the weights of the BiLSTM-CRF network.

Once the model has been trained, the prediction of a BiLSTM-CRF is the sequence of labels maximizing the model for the given the input sequence and the learned parameters (Equation 2.34).

$$y^* = \arg\max_{y} p(y|\mathbf{X}, \theta, \mathbf{A}) \tag{2.34}$$

The labels are typically predicted using dynamic programming algorithms such as Viterbi which provides the optimal prediction for the measurement sequence as a whole.

Building upon the BiLSTM-CRF, researchers have proposed many alternatives to this network. For example, Lample et al. [2016] have proposed to add a character-level word encoding to the word embeddings to learn the morphological patterns in the words; Ma and Hovy [2016] have used a convolutional neural network (CNN) for the character-level word embedding; Peters et al. [2018] have used pre-trained LSTM-based language models to learn contextual word embeddings; and Devlin et al. [2019] have replaced the LSTM encoder for more efficient pre-trained transformer. However, in most of the state-of-the-art models the underlying architecture is very similar to the BiLSTM-CRF.

2.3.4.2 Sequence-to-Sequence Models

Sequence-to-sequence (*seq2seq*) models are the base architecture for neural machine translation (NMT) models. Seq2seq has revolutionized the area of MT, providing a deep-learning framework that can be trained end-to-end and that has improved considerably the translation quality of previous existing models such as the PBSMT [Bojar *et al.*, 2016; Bojar *et al.*, 2017; Sutskever *et al.*, 2014]. It is ubiquitously used in learning tasks where the lengths of the input and output sequences are different (as such, it is not commonly used in NER).

Seq2seq models are based on the encoder-decoder architecture (see Figure 2.12). Usually, both the encoder and the decoder are some type of neural network such as the LSTM [Sutskever *et al.*, 2014], GRU or transformer [Vaswani *et al.*, 2017]. In any case, first the

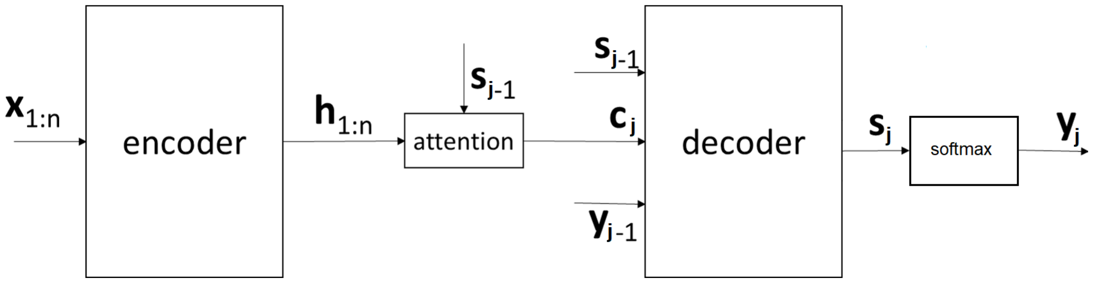


Figure 2.12: Typical encoder-decoder architecture of a seq2seq model.

encoder converts the input sequence of word embeddings $\mathbf{x}_i$ into a sequence of hidden vectors $\mathbf{h}_i$:

$$\mathbf{h}_i = encoder(\mathbf{x}_i, \mathbf{h}_{i-1}) \quad i = 1, \ldots n \tag{2.35}$$

where n is the number of tokens in the input sentence. Then, given all the hidden vectors, a fixed dimensional context vector $\mathbf{c}_j$ is defined for each decoding step j. The context vector, is computed as a weighted sum of the hidden vectors:

$$\mathbf{c}_j = \sum_{i=0}^{n} \alpha_{ij} \mathbf{h}_i \qquad j = 1, \ldots m \tag{2.36}$$

where α_{ij} are the attention weights and m is the total number of decoding steps (the number of tokens in the target sentence). Many different algorithms have been proposed to compute the attention weights such as the additive attention [Bahdanau *et al.*, 2015], the multiplicative attention [Luong *et al.*, 2015] or the multi-headed attention usually used in transformers [Vaswani *et al.*, 2017]. Then, the decoder generates m hidden vectors one by one using the context words, the previous target word's embedding $\mathbf{y}_{j-1}$ and the previous decoding hidden vector $\mathbf{s}_{j-1}$:

$$\mathbf{s}_j = decoder(\mathbf{c}_j, \mathbf{s}_{j-1}, \mathbf{y}_{j-1}) \qquad j = 1, \ldots m \tag{2.37}$$

Finally, the hidden vectors $\mathbf{s}_j$ are propagated through a linear layer and a softmax layer in order to output the probability distribution of the words in the target vocabulary:

$$\mathbf{p}_j = softmax(\mathbf{W}\mathbf{s}_j + \mathbf{b}) \qquad j = 1, \ldots m \tag{2.38}$$

where **W** and **b** are trainable parameters. p_j is a vector with the dimension of the same size as the target vocabulary. During training, the seq2seq model is trained to maximize the probability of the words in the ground-truth sentence with the negative log-likelihood (NLL) loss. Additionally, the common practice is to use a "teacher-forcing" setting, where the embeddings of the previous words in the decoding steps are taken from the ground-truth reference and not from the model's own predictions. This is done to speed-up training convergence.

However, during inference, the words of the ground-truth are not available. As usual in other discriminative models, the optimal y sequence that maximizes the probability of the translation needs to be found:

$$y^* = \arg\max_y p(y|x, \theta) \tag{2.39}$$

where x is the source sentence and y is its translation. But the output label space is too large in MT to be able try all the possible translation. Thus, researchers have proposed other more efficient sub-optimal inference algorithms such as *beam search* [Sutskever *et al.*, 2014]. The beam search algorithms predicts the most probable B words at the first decoding step. After that, it continues decoding the most likely next words as a tree search, discarding all paths except the most probable B paths. The search is stopped when the end-of-sentence (<EOS>) is predicted. Figure 2.13 shows how beam search predicts a translation hypothesis.

Currently, the state-of-the-art NMT model is that proposed by Vaswani et al. [2017] and is based on transformer encoder and decoders (see Figure 2.14) [Bojar *et al.*, 2018; Barrault *et al.*, 2019]. See Section 2.3.5 for more details on the Transformer

2.3.5 Transformer

As mentioned in the previously, RNNs can encode sequences of tokens from left-to-right and right-to-left. Yet, to obtain each hidden vector in the sequence, the hidden vector of the previous element needs to be already computed. This inherent sequential nature of RNNs does not allow for parallelization. Thus, Vaswani et al. [2017] have proposed the Transformer, a neural network architecture that avoids any recurrent connection and

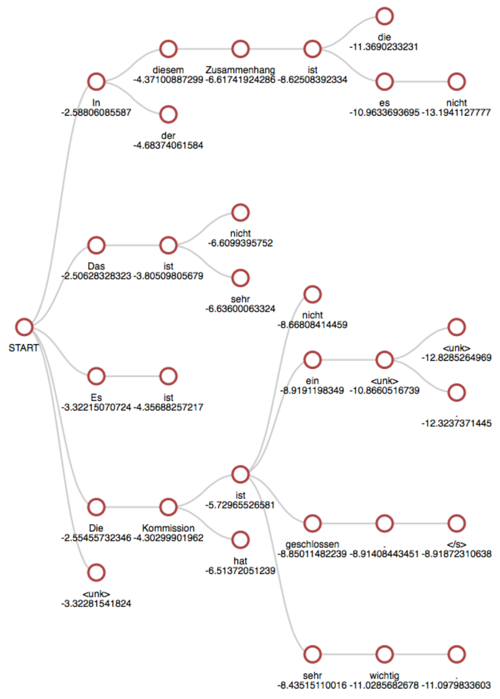


Figure 2.13: Beam search with beam size $B = 5$ in a English-German translation example. Image taken from OpenNMT[2]

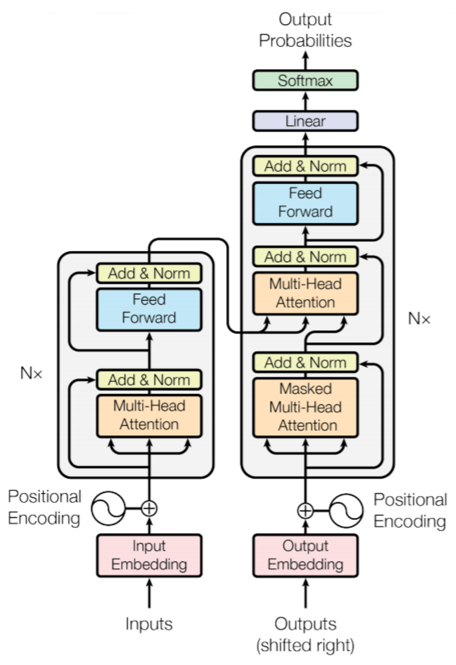


Figure 2.14: Transformer based encoder-decoder model from [Vaswani *et al.*, 2017] . (left) encoder and (right) decoder. The multi-head attention on the decoder uses the hidden vectors from the decoder as keys and queries to learn the attention weights.

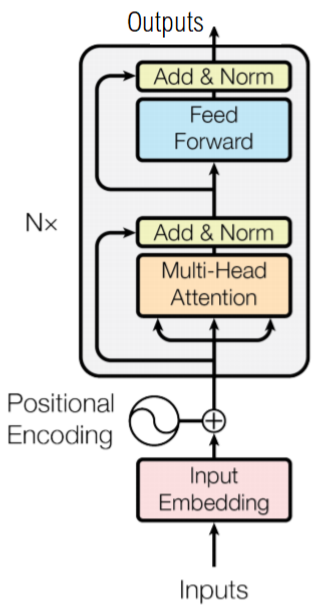


Figure 2.15: Transformer encoder [Vaswani *et al.*, 2017].

relies solely on the attention mechanism to learn the sequential dependencies of the tokens. This architecture is highly parallelizable and, consequently, more computationally efficient, reducing considerably the convergence time of the model. Figure 2.15 shows the architecture of a single layer of the Transformer sequence encoder.

First, the tokens of the input sentence $(x_1, x_2, \ldots, x_n)$ are encoded with their corresponding word embeddings $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$. An additional positional encoding is added to the word vector to make use of the order of the words in the sequence. The positional encoding has the same dimension as the word embedding (d_{model}) so that both can be added and it is obtained with two sinusoidal functions of different frequencies:

$$\begin{aligned} \mathbf{PE}_{(pos,2i)} &= \sin(pos/10000^{2i/d_{model}}) \\ \mathbf{PE}_{(pos,2i+1)} &= \cos(pos/10000^{2i/d_{model}}) \end{aligned} \tag{2.40}$$

where pos is the position of the word in the sequence and i is the dimension in the vector. The resulting word encoding is:

$$\mathbf{e}_{pos} = \mathbf{x}_{pos} + \mathbf{PE}_{pos} \qquad pos = 1, \ldots, n \tag{2.41}$$

Then, a multi-headed self-attention mechanism models the intra-sequence dependencies between tokens. This module first maps the input embeddings $\mathbf{e}_{pos}$ of dimension

d_{model} into a *query* (Q), *key* (K) and *value* (V) vectors of dimension $d_k = d_{model}/h$, being h the number of heads. Lets say that $\mathbf{E} \in \mathbb{R}^{n \times d_{model}}$ is the input matrix that contains all the n embeddings of the tokens in the input sentence, then the three mappings are obtained:

$$\begin{aligned} \mathbf{Q} &= \mathbf{E}\mathbf{W}^Q \\ \mathbf{K} &= \mathbf{E}\mathbf{W}^K \\ \mathbf{V} &= \mathbf{E}\mathbf{W}^V \end{aligned} \tag{2.42}$$

where $\mathbf{W}^Q \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}^K \in \mathbb{R}^{d_{model} \times d_k}$ and $\mathbf{W}^V \in \mathbb{R}^{d_{model} \times d_k}$ are learnable parameters. Given these three mappings, the attention mechanism computes the new hidden vectors stored in matrix $\mathbf{Z} \in \mathbb{R}^{n \times d_k}$. The authors used the multiplicative attention [Luong *et al.*, 2015] instead of the additive attention [Bahdanau *et al.*, 2015] because it is computationally more efficient:

$$\mathbf{Z} = Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax(\frac{QK^T}{\sqrt{d_k}})\mathbf{V} \tag{2.43}$$

But the authors have shown that it is beneficial to use many different attention mechanisms in parallel rather than a single one. Therefore, they have proposed to learn different linear projections $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$ and $\mathbf{W}_i^V$. The multi-headed attention is computed:

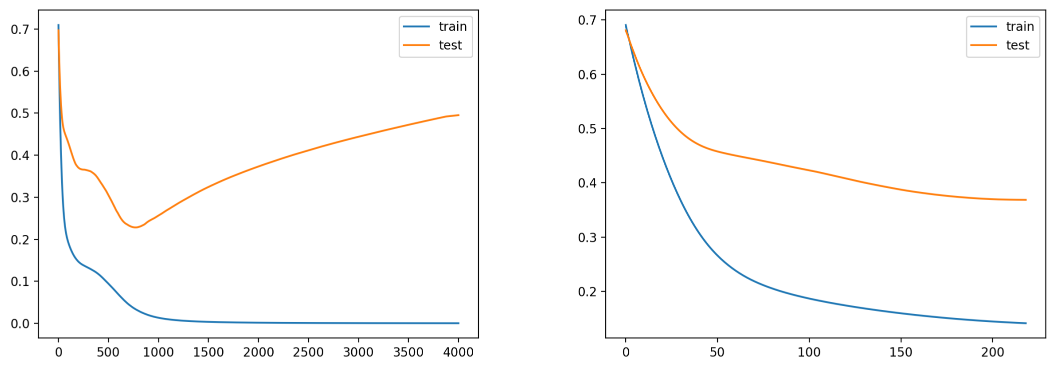
$$MultiHead(\mathbf{E}) = Concat(head_1, head_2, \ldots, head_h)\mathbf{W}^O \tag{2.44}$$

where $head_i = Attention(\mathbf{E}\mathbf{W}_i^Q, \mathbf{E}\mathbf{W}_i^K, \mathbf{E}\mathbf{W}_i^V)$ and $\mathbf{W}^O \in \mathbb{R}^{hd_k \times d_{model}}$ are learnable parameters. The authors also include a residual connection [He *et al.*, 2016] and layer normalization [Ba *et al.*, 2016] to ease the training of the network.

$$\mathbf{Z} = LayerNorm(\mathbf{E} + MultiHead(\mathbf{E})) \tag{2.45}$$

Finally, the transformer propagates the output of the self-attention sublayer into a feed-forward network, which also uses residual connections and layer normalization (see Figure 2.15).

This is a single layer of the transformer, but usually in NLP many layers are stacked together when encoding sequences. For a more detailed description of the transformer



(a) Overfitted model (b) Not overfitted model

Figure 2.16: Performance of a model over a training (bleu) and test (orange) sets.

we refer the reader to the *Ilustrated Transformer*[3]. The transformer has become the most common neural network encoding/decoding model on NER and MT tasks.

2.4 Low-Resource Deep Learning

The previous sections have described the state-of-the-art sequential labeling models which nowadays are predominantly based on deep learning. However, this only holds when there is enough amount of supervised data to properly train the models, as it has been shown that neural networks suffer significantly from *overfitting* [Lawrence *et al.*, 1997]. Overfitting means that the model has learned to fit the training data "too well" and is not able to generalize to new unseen data. In a simple analogy, it's as if a kid memorized all the answers of his/her maths homework, but then could not answer correctly a new question in the exam. In essence, the kid did not understand the task. Thus, an overfitted model would perform like in Figure 2.16a, where the training error would be almost zero, but the test error would start growing after a point. While in a properly trained model, the test error should not start growing after certain iterations even if the training error is not zero (Figure 2.16b).

When there is a lot of properly annotated, good quality training data, it is easier to avoid overfitting, as the model is trained over many examples and becomes more robust. However, as mentioned before, abundant supervised data are often not available in many

[3] http://jalammar.github.io/illustrated-transformer/

NLP tasks. Consequently, researchers have become very interested in developing regularization and training techniques that can help the model to generalize better on different NLP tasks, using fewer annotated data. This has been the main motivation of this thesis, too. This section covers some of the most popular approaches that have been proposed for improving the generalization of deep neural networks to date.

2.4.1 Early Stopping

Early stopping [Prechelt, 1998] is a widespread regularization technique in deep learning. The approach is very simple, yet very effective. It consists of monitoring the performance of the model over the validation set during training and terminating training accordingly.

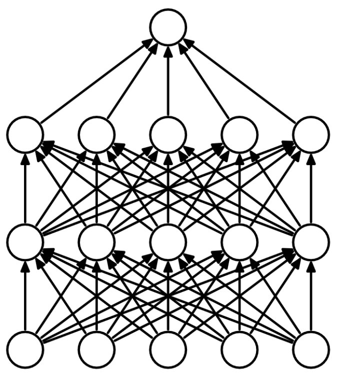
Typically, the training data are first divided into a training and a validation sets (e.g. 70/30). Then, the model training commences solely over the training data. Every certain number of batches, the model is evaluated over the validation set. If the evaluation score between subsequent validation turns meets the stopping criteria, the training is immediately terminated. A stopping criterion, for example, can be that the new validation score is lower than the previous one.

The effectiveness of early stopping can be observed by looking at Figure 2.16b. With the stopping criterion mentioned in the previous paragraph, the model would have stopped training around batch 750, and it would have avoided overfitting. Early stopping has become a common regularization technique in NER and MT deep learning based models.

2.4.2 Dropout

Dropout [Srivastava *et al.*, 2014] is another very effective regularization technique with deep neural networks. As mentioned by the authors of the original paper, one of the best ways to regularize a neural network is to "ensemble" the predictions of multiple different neural networks. However, this is most often infeasible, as doing inference for multiple neural networks can be very time-consuming. Dropout, instead, is used over a single model.

The idea in dropout is to randomly remove connections from the neural network at each training iteration (see Figure 2.17b). Thus, at each training step, a different "thinned"


(a) Standard Neural Net

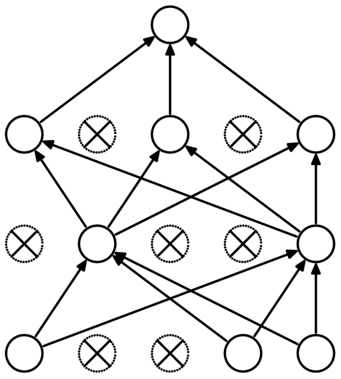

(b) After applying dropout

Figure 2.17: Full model vs dropout.

version of the whole neural network is trained. The practical way of doing this is by randomly multiplying some weights with zero. The percentage of the weights that are set to zero is a tuneable hyper-parameter, but usual values oscillate between 0.3 and 0.5. Dropout has been proven to be a very powerful technique to avoid overfitting and, thus, different variations of it have been proposed in recent years [Ba and Frey, 2013; Gal and Ghahramani, 2016].

Finally, it is important to say that even if in most cases dropout helps to improve the accuracy of the model, it can sometimes have the opposite effect and damage the model, as shown by [Lan *et al.*, 2020]. Their experiments show that when the transformer-based ALBERT model has been trained with sufficiently large amounts of training data, the downstream tasks (e.g. NER) accuracy is higher when the model does not use dropout. Nevertheless, dropout is commonly used in NER and MT deep learning based models.

2.4.3 Data Augmentation

Overfitting is more prevalent in small datasets. Therefore, many researchers have looked into ways of increasing the size of the datasets in a fast and economic way, without the need for annotation from human experts. Datasets of this kind often get the name of *silver-corpora*.

In NER, a common approach is to use ‘bootstrapping’ [Nadeau and Sekine, 2007], which is a semi-supervised learning (SSL) technique. The model usually starts with a small list of annotated names. It parses the corpus looking for mentions of those names and tries to extract some contextual clues that surround those names. Then, the model looks for new instances that co-occur in the same contexts, and the list of entities is expanded. This process can be repeated until a large amount of named entities is gathered. Multiple SSL learning approaches for NER have been proposed in recent years [Brin, 1998; Collins and Singer, 1999; Yangarber *et al.*, 2002]. Training supervised models over these collected entities can contribute to improve the model.

Many ways of augmenting the training data have also been proposed for MT. A common approach is to crawl the web looking for bilingual or even multilingual web pages that contain *bitexts* [Harris, 1988], mutual translations in different languages. Esplà-Gomis et al. [2014] have distinguished three main approaches of crawling the web for parallel texts: finding similarities in the URLs [Ma and Liberman, 1999; Chen *et al.*, 2004; Espla-Gomis and Forcada, 2010; San Vicente *et al.*, 2012], exploiting parallel structures of HTML files [Espla-Gomis and Forcada, 2010; San Vicente *et al.*, 2012; Papavassiliou *et al.*, 2013] and content-similarity techniques based on bag-of-words [Ma and Liberman, 1999; Antonova and Misyurev, 2011; Barbosa *et al.*, 2012].

Another strategy to obtain additional data in MT is to infer back-translations from a monolingual corpus. For example, Sennrich et al. [2016] use a large monolingual dataset in the target language and a pre-trained MT model to translate those sentences from *target$\rightarrow$source*. Training an NMT model with these additional data helps to improve its decoder's side language model and, as a result, the overall BLEU score. Other researchers have also proposed different ways to perform back-translations [Poncelas *et al.*, 2018; Hoang *et al.*, 2018; Edunov *et al.*, 2018a; Fadaee and Monz, 2018].

Finally, inspired by techniques developed in computer vision, Fadaee et al. [2017] have proposed to create synthetic translation samples. In computer vision, synthetic samples of images are created by rotating or flipping the images. For NMT, the authors have proposed to target low-frequency words in order to generate new sentence pairs containing these rare words. Figure 2.18 shows an example.

2.4.4 Multi-task learning

In recent years, multi-task learning (MTL) has become another very effective strategy to improve the ability of deep learning models to generalize, with widespread applications

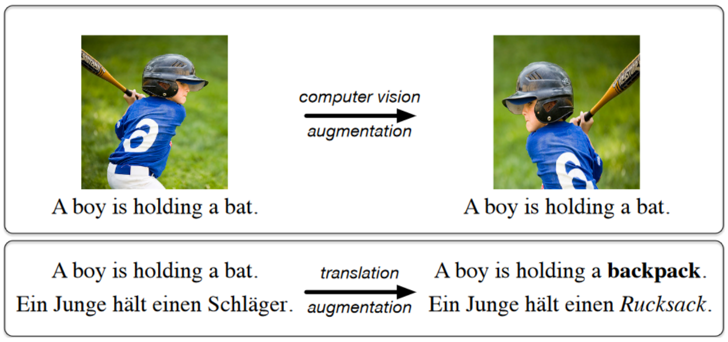
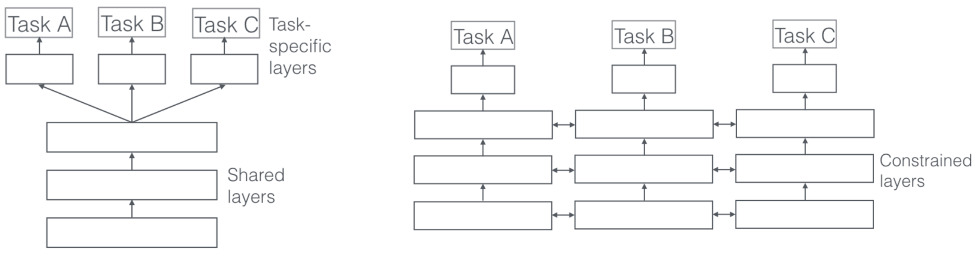


Figure 2.18: Data augmentation in computer vision and NMT [Fadaee *et al.*, 2017]. Word *bat* is replaced by word *bag*, creating a new synthetic sentence that is grammatically and semantically correct (based on the probability accorded by a trained language model).



(a) Hard parameter sharing

(b) Soft parameter sharing

Figure 2.19: Two different MTL strategies.

in fields such as NLP [Collobert and Weston, 2008], computer vision [Girshick, 2015], speech recognition [Deng *et al.*, 2013] and drug discovery [Ramsundar *et al.*, 2015], among others. The underlying principle is to use multiple loss functions or "tasks", which are different yet related, forcing the neural model to learn shared representations that can perform well in all of them (and, possibly, others).

Ruder [2017] classifies MTL in two main approaches: *hard parameter sharing* and *soft parameter sharing*. In the former, several layers of the neural networks are shared between all tasks, and only the output layer is different between them (see Figure 2.19a from [Ruder, 2017]). In the latter, each task has its own model with its own parameters, but the parameters of the different models are regularized by encouraging them to be similar (see Figure 2.19b from [Ruder, 2017]). For example, Duong et al. [2015] use ℓ_2 distance and Yang and Hospedales [2016] use the trace norm for regularization.

Aguilar et al. [2017] have recently proposed an improved multi-task neural architecture for NER in social media. They separate entity categorization and entity segmentation in two tasks using two objective functions. Crichton et al. [2017] have used a multi-task convolutional neural network that is jointly trained in many different biomedical NER tasks (with different classes) and showed that the model can improve results most of the times. However, MTL can also damage the performance of the models. Some researchers [Changpinyo *et al.*, 2018; Søgaard and Goldberg, 2016] have shown the performance of NER models drops if it is trained jointly with tasks such as POS tagging, chunking, sentence comprehension, surpersense tagging or semantic tagging.

MTL has also been used in MT. Zaremoodi and Haffari [2018] have used monolingual linguistic resources on the source side in a MTL approach. The auxiliary tasks used include NER, semantic parsing and syntactic parsing. Their experiments have showed substantial improvements in low-resource language pairs. Similarly, Zhang and Zong [2016] have used the source sentence information and train the model to learn to reorder the source sentence as an auxiliary task. Eventually, Domhan and Hieber [2017] have leveraged target side information in an MTL setting, using a language modelling auxiliary task in order to learn to translate better.

2.4.5 Sequence-level training

A major source of overfitting in NMT models is the *exposure bias* [Bengio *et al.*, 2015]. NMT models usually operate at token-level, meaning the classifier is trained to maximize the probability of the tokens in the translation, token-by-token, left to right. During training, the model uses "teacher forcing", which means the decoder uses the previous token from the reference sentence to infer the probability of the current token. However, at test time, the reference sentence is not available, and the model needs to rely on its own previous token predictions to decode the sentence token by token. This training-inference discrepancy is known as exposure bias and can seriously damage the performance of the model at test time.

In order to solve this problem, researchers have proposed to train NMT and other natural language generation (NLG) models using sequence-level objective functions, with reinforcement learning style training [Ranzato *et al.*, 2015; Paulus *et al.*, 2018]. One of the most widespread algorithms used for sequence-level training is REINFORCE [Sutton and Barto, 2018], which defines the loss function as the negative expected reward (r) of a policy ($p(y|x;\theta)$):

$$\mathcal{L}(y) = -\mathbb{E}[r(y)]_{p(y|x;\theta)} \approx \sum_{\hat{y}\sim p(y|x;\theta)} -r(\hat{y})p(\hat{y}|x;\theta) \tag{2.46}$$

in NMT the policy is a seq2seq neural network, and the expectation is approximated with few $\hat{y}$ sentences (e.g. $1 \sim 5$), which are typically obtained by either sampling or beam search. This objective function uses the model's own predictions ('trajectories') during training, therefore, the exposure bias is removed.

Moreover, the REINFORCE algorithm allows us to minimize any loss function (or maximize any reward function) that cannot be differentiated in θ. In NLG tasks, models can be directly trained in typical, discontinuous evaluation functions such as BLEU, METEOR, TER or ROUGE. The target function is defined as the reward in Equation 2.46. Then, ignoring the indirect dependence of r on θ and using the policy gradient theorem, we can calculate the derivative of the objective function and backpropagate gradients through the policy network:

$$\begin{aligned}\frac{\partial}{\partial\theta} - \mathbb{E}[r(y)]_{p(y|x;\theta)} &= -\mathbb{E}[r(y)\frac{\partial}{\partial\theta}\ln p(y|x;\theta)]_{p(y|x;\theta)} \\ &\approx \sum_{\hat{y}\sim p(y|x;\theta)} -r(\hat{y})p(\hat{y}|x;\theta)\frac{\partial}{\partial\theta}\ln p(\hat{y}|x;\theta)\end{aligned} \tag{2.47}$$

Note the simple rule from calculus ($\frac{\partial \ln f(u)}{\partial u} = \frac{1}{f(u)}\frac{\partial f(u)}{\partial u} \longrightarrow \frac{\partial f(u)}{\partial u} = f(u)\frac{\partial \ln f(u)}{\partial u}$) that is applied in Equation 2.47.

In practice, due to the prediction space in MT and because the expectation is approximated with very few samples, the REINFORCE algorithm has a very large variance and often fails to converge. In order to reduce the variance of the model, researchers have proposed different approaches such as reward with baseline or pre-training NMT models with the NLL loss to avoid a "cold start" [Sutton and Barto, 2018]. Edunov et al. [2018b] have experimented with various alternative variations of the REINFORCE algorithm to train NMT models. Fine-tuning with these sequence-level losses has proven to be beneficial, particularly in low-resource language pairs.

2.4.6 Transfer Learning

Let us conclude this chapter talking about transfer learning, which has become a very popular approach for many low-resource NLP tasks. In transfer learning, a neural network is first pre-trained in a high resourced supervised dataset or in a large unsupervised dataset. In this pre-training stage, the model is normally trained in a very general task. Then, this model is used as basis to fine-tune the model for a more-specific, low-resource task. Word embeddings such as word2vec, GloVe or fastText are an example of such transfer learning. However, recently researchers have proposed more sophisticated transfer learning methods.

One popular approach is to use pre-trained language models such as ELMo [Peters *et al.*, 2018], GPT[Radford *et al.*, 2018], BERT [Devlin *et al.*, 2019] or XLNet [Yang *et al.*, 2019]. These are very large deep networks, based on RNNs or transformers, that leverage large amounts of unsupervised data to learn robust language models. Usually, a language model learns the probability of a sentence $(t_1, \ldots, t_N)$ by modelling the conditional probability of each token t_i in the sentence given the preceding tokens:

$$p(t_1, \ldots, t_N) = \prod_{i=1}^{N} p(t_i | t_1, \ldots, t_{i-1}) \tag{2.48}$$

where N is the number of tokens in the sentence. Typically, bidirectional language models model the sentences *left-to-right* and *right-to-left*. However, Devlin et al. [2019] have proposed an alternative method to learn deep bidirectional language models that has obtained very promising results. They call it the *masked language model* (MaskLM) and is inspired by the *cloze* task, as they randomly mask a percentage of words in the sentence and the network needs to learn to predict the masked tokens given the unmasked tokens (preceding and future). Different from the de-noising autoencoders [Vincent *et al.*, 2008], in MaskLM the model only has to predict the masked tokens and not reconstruct the whole sentence.

Once the language models have been pre-trained, either with a normal LM or a Masked LM, there are two main ways of using them for transfer learning in less resourced tasks: a *feature-based* approach and a *fine-tuning* approach. On the one hand, feature-based approaches use task specific architectures such as the BiLSTM, and add the pre-trained

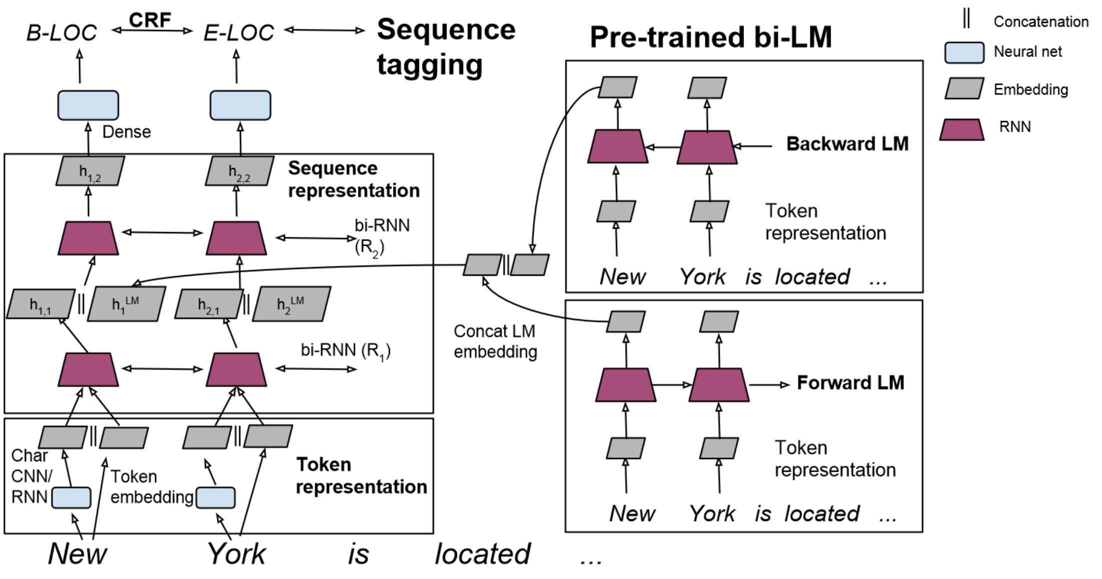


Figure 2.20: A feature-based approach using ELMo LM representations [Peters *et al.*, 2017]. The vectors learned from the ELMo network are used as features in a task-specific architecture, in this case, a two-layer BiLSTM. ELMo vectors are concatenated with the output of the first LSTM layer.

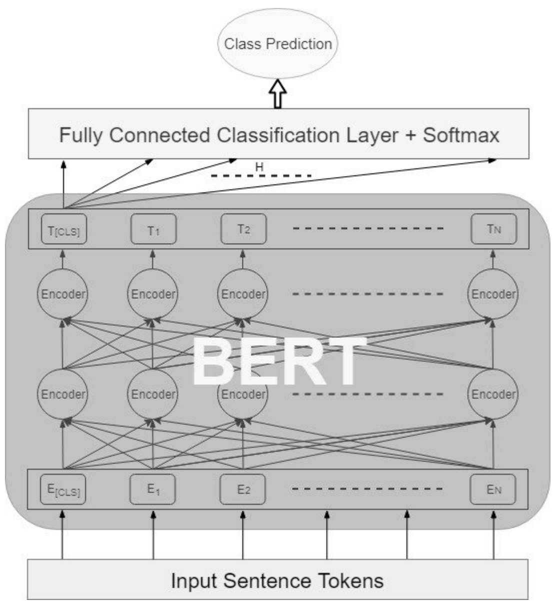


Figure 2.21: A fine-tuning approach using BERT [ElJundi *et al.*, 2019]. A classification layer is adde to the BERT architecture. The whole network is fine-tuned with a task-specific loss.

representations as additional features. ELMo [Peters *et al.*, 2018] uses this approach (see Figure 2.20). On the other hand, fine-tuning approaches do not use many task-specific parameters (e.g. just add an output layer), but fine-tune all the pre-trained parameters on the low-resource task. BERT [Devlin *et al.*, 2019] has been conceived to be used in this way (see Figure 2.21).

Pre-training language models have achieved state-of-the-art results in many low resourced NER tasks. Just as an example, Liu et al. [2018] have recently proposed to combine pre-trained language models and multi-tasking, reporting significant accuracy improvements.

In MT, few researchers have aimed to use pre-trained language models, but the results have not been as promising. Edunov et al. [2019] have used deep contextualized embeddings as input to the encoder and the decoder. They have showed that these representations have been useful in the encoder, but not in the decoder. Alternatively, researchers have proposed another way to perform transfer learning in NMT models. The main idea is to pre-train robust NMT models over high-resource language pairs such as French-English or Spanish-English, and then fine-tune those models over less resourced language pairs such as Basque-English. Zoph et al. [2016] have used this approach, obtaining up to 5.6 BLEU points of improvement over four low-resource language pairs.

Chapter 3

Recurrent Neural Networks with Specialized Word Embeddings for Health-Domain Named-Entity Recognition

3.1 Introduction

In the healthcare system, patients' medical records represent a big data source. Even though the records contain very useful information about the patients, in most cases the information consists of unstructured text such as, among others, doctors' notes, medical observations made by various physicians, and descriptions of the recommended treatments. This type of data cannot be analyzed using common statistical tools; rather, they need to be approached by Natural Language Processing (NLP) techniques. Health-domain NER aims to automatically find "named entities" in text and classify them into predefined categories such as people, locations, hospitals, drugs, brands etc. In the health domain, the two most important NER tasks are Clinical Concept Extraction (CCE) and Drug Name Recognition (DNR). The former aims to identify mentions of clinical concepts in patients' records to help improve the organization and management of healthcare services. Named entities in CCE can include test names, treatments, problems related to individual

Sentence	*the*	*effects*	*of*	*chronic*	*phenyotin*	*or*	*carbamazepine*	*therapy*
Entity class	*O*	*O*	*O*	*O*	*B-drug*	*O*	*B-drug*	*O*

a) DNR example

Sentence	*his*	*lateral*	*percutaneous*	*drains*	*had*	*been*	*pulled*	*out*
Entity class	*B-treatment*	*I-treatment*	*I-treatment*	*I-treatment*	*O*	*O*	*O*	*O*

b) CCE example

Figure 3.1: (a) DNR and (b) CCE tasks examples, where ‘B’ (beginning) specifies the start of a named entity, ‘I’ (inside) specifies that the word is part of the same named entity, and ‘O’ (outside) specifies that the word is not part of any predefined class.

patients, and so forth. The latter seeks to find drug mentions in unstructured biomedical texts to match drug names with their effects and discover drug-drug interactions (DDIs). DNR is a key step of pharmacovigilance (PV) which is concerned with the detection and understanding of adverse effects of drugs and other drug-related problems. Figure 3.1 shows examples of both tasks.

NER is a challenging learning problem because in most domains the training datasets are scarce, preventing a “brute-force” approach by exhaustive dictionaries. Consequently, many systems rely on handcrafted rules and language-specific knowledge to solve this task. To give a simple example of such rules, if the word begins with a capital letter in the middle of the sentence, it can be assumed to be a named entity in most cases. Nevertheless, these approaches are time-costly to develop, depend considerably on the language and the domain, are ineffective in the presence of informal sentences and abbreviations and, although they usually achieve high precision, suffer from low recall (i.e., they miss many entities). Conversely, machine learning (ML) approaches overcome all these limitations as they are intrinsically robust to variations. More modern ML methods follow a two-step process: (1) feature engineering and (2) automated classification [Segura-Bedmar *et al.*, 2015; Abacha *et al.*, 2015; Rocktäschel *et al.*, 2013; de Bruijn *et al.*, 2011]. The first step represents the text by numeric vectors using domain-specific knowledge. The second step refers to the task of classifying each word into a different named-entity class, with popular choices for the classifier being the linear-chain CRFs, Structural Support Vector Machines (S-SVM) and maximum-entropy classifiers. The drawback of this approach is that feature engineering can be often as time-consuming as the manual design of rules.

In recent years, the advent of deep learning has contributed to significantly overcome the feature engineering problem in ML. More specifically, in general domain NER the Long Short-Term Memory (LSTM) and its variants (e.g., the Bidirectional LSTM) have reported very promising results [Lample *et al.*, 2016]. In these models, unique words only need to be initialized with random vectors, and during training the neural network is able to automatically learn improved representations for them, completely bypassing feature engineering. In order to further increase the performance of these systems, the input vectors can alternatively be assigned with general-purpose word embeddings learned with GloVe [Pennington *et al.*, 2014], Word2vec [Mikolov *et al.*, 2013b] or other algorithms. The aim of general-purpose word embeddings is to map every word in a dictionary to a numerical vector (the embedding) so that the distance between the vectors somehow reflects the semantic difference between the words. For example, 'cat' and 'dog' should be closer in the vector space than 'cat' and 'car'. The common principle behind embedding approaches is that the meaning of a word is conveyed by the words it is used with (its surrounding words, or context). Therefore, the training of the word embeddings only requires large, general-purpose text corpora such as Wikipedia (400 K unique words) or Common Crawl (2.2 M unique words), without the need for any manual annotation. As well as semantic word embeddings, character-level embeddings of words can also be automatically learned [Lample *et al.*, 2016]. Such embeddings can capture typical prefixes and suffixes, providing the classifiers with richer representations of the words.

However, as mentioned before, drug and clinical concept recognition are very domain-specific tasks, and supervised datasets for training state-of-the-art RNNs are often scarce. Even the training of the word embeddings is challenged by the fact that general-purpose text corpora often do not contain the technical words and specialized concepts that are inherent in the health domain. In order to assign word embeddings to such specialized words, the embedding algorithms should be retrained using medical domain resources such as, for instance, the MIMIC-III corpora [Johnson *et al.*, 2016a].

[Chalapathy *et al.*, 2016b] have obtained very promising accuracy with a DNR system that uses a Bidirectional LSTM-CRF architecture with random assignments of the input word vectors at the EMNLP 2016 Health Text Mining and Information Analysis workshop. The reported results were very close to the system that ranked first in the SemEval-2013 Task 9.1. [Chalapathy *et al.*, 2016a] have leveraged the same architecture for CCE at the Clinical NLP 2016 workshop, this time using pre-trained word embeddings from GloVe, and the results outperformed previous systems over the i2b2/VA dataset.

In this chapter, we evaluate if these systems can be further improved by training the deep networks with more complex and specialized word embeddings. Moreover, the impact of augmenting the word embeddings with conventional feature engineering is explored. As methods, we compare contemporary recurrent neural networks such as the Bidirectional LSTM and the Bidirectional LSTM-CRF against a conventional ML baseline (a CRF). We report state-of-the-art results in both DNR and CCE, outperforming systems that follow the traditional two-step machine learning approach.

3.2 Related work

Most of the pre-deep-learning research carried out in domain-specific NER has combined supervised and semi-supervised ML models with text feature engineering. This is the case, for example, for most of the systems that participated on the SemEval-2013 Task 9.1 [Herrero-Zazo *et al.*, 2013], a shared-task organized for the development of NER systems targeting the recognition and classification of names of pharmacological substances (i.e. drug name recognition (DNR)). WBI-NER [Rocktäschel *et al.*, 2013], the system that ranked first in the shared, is based on a linear-chain CRF with a combination of general and domain specialized features. General domain features include whether the token is part of a sentence with all uppercase letters, the text of four preceding and succeeding tokens, and other common morphological, syntactic and orthographic features proposed in the literature [Klinger *et al.*, 2008; Settles, 2005; Leaman and Gonzalez, 2008]. Domain specific features include the use of ChemSpot [Rocktäschel *et al.*, 2012] to check whether a token is part of a chemical; to check whether the token appears in the Jochem [Hettne *et al.*, 2009] chemical dictionary, or ontology-based features that can help to distinguish whether a entity refers to a specific drug or a generic term denoting a group of drugs [De Matos *et al.*, 2010; Coulet *et al.*, 2011]. Later work on DNR have followed similar approaches to the ones propsed by WBI-NER [Abacha *et al.*, 2015;

Liu *et al.*, 2015].

In CCE, a similar approach (general/domain specific feature engineering + conventional ML classifier) has achieved the best results [de Bruijn *et al.*, 2011; Boag *et al.*, 2015]. For example, the previous state-of-the-art proposed by de Bruijn et al. [2011] have trained a a semi-Markov HMM, which can tag multitoken spans of text, using a combination of general domain features such as character n-grams, token n-grams, sentence length indicators, section features (e.g. headings, subsections, paragraphs), and domain specific external annotation tools such as cTakes [Savova *et al.*, 2010], MetaMap [Aronson and Lang, 2010] or ConText [Harkema *et al.*, 2009]. Insipired by Miller et al. [2004], they have also used the Brown clustering algorithm [Brown *et al.*, 1992].

In health-domain, pretrained word embeddings [Pennington *et al.*, 2014; Mikolov *et al.*, 2013b; Lebret and Collobert, 2013] have been used in traditional ML methods [Nikfarjam *et al.*, 2015; Wu *et al.*, 2015] and in neural networks, where [Dernoncourt *et al.*, 2017] have achieved better performance than previously published systems in de-identification of patient notes. [Cocos *et al.*, 2017] have used the Bidirectional LSTM model for labelling Adverse Drug Reactions in pharmacovigilance. [Xie *et al.*, 2017] have used a similar model for studying the adverse effects of e-cigarettes. [Wei *et al.*, 2016] have combined the output of a Bidirectional LSTM and a CRF as input to an SVM classifier for disease name recognition. A possible drawback of this approach is that the overall prediction is not structured and may miss on useful correlation between the output variables.

In a work that is more related to ours, [Jagannatha and Yu, 2016] have employed a Bidirectional LSTM-CRF to label named entities from electronic health records of cancer patients. Their model differs in the CRF output module where the pairwise potentials are modelled using a Convolutional Neural Network (CNN) rather than the usual transition matrix. [Gridach, 2017] has also used the Bidirectional LSTM-CRF for named-entity recognition in the biomedical domain. The main difference and contribution of our proposed approach is that it leverages specialized health-domain embeddings created from a structured database. In the experiments, these embeddings have been used jointly with general-domain embeddings and they have proved able to improve the accuracy in several cases. In addition, our work evaluates the use of hand-crafted features in the system [Lee *et al.*, 2016]. This aims to provide a comprehensive feature comparison for health-domain named-entity recognition based on LSTM models.

3.3 Methods

3.3.1 CRF

We train a CRF model as a useful baseline for performance comparison with the proposed neural networks. For its implementation, we have used the HCRF library[1]. The features used as input are described in Section 3.4. Note that a CRF model is also used as the output layer in the Bidirectional LSTM-CRF.

3.3.2 Bidirectional LSTM and bidirectional LSTM-CRF

For the RNN approach, we train two models: 1) The Bidirectional LSTM (B-LSTM) and 2) the Bidiretional LSTM-CRF (B-LSTM-CRF), both described in Chapter 2. However, we have used a different implementation of the LSTM unit proposed by [Lample *et al.*, 2016], the state-of-the-art NER model at the time of conducting this research (i.e. 2017), and showed in Equation 3.1. Note that in this implementation the forget gate ($\mathbf{f}_t$) is missing and that both the input gate ($\mathbf{i}_t$) and the output gate ($\mathbf{o}_t$) depend directly from the memory cell state ($\mathbf{c}$).

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{V}_i\mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{c}_t &= (1 - \mathbf{i}_t) \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{V}_o\mathbf{c}_{t-1} + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \circ \tanh(\mathbf{c}_t)
\end{aligned} \tag{3.1}$$

We test the LSTM models with the same features used for the CRF in order to establish the fairest-possible comparison. The features are described in detail in Section 3.4. Figure 3.2 shows a descriptive diagram of the Bidirectional LSTM-CRF.

[1]HCRF. Available from: <http://multicomp.ict.usc.edu/?p=790>

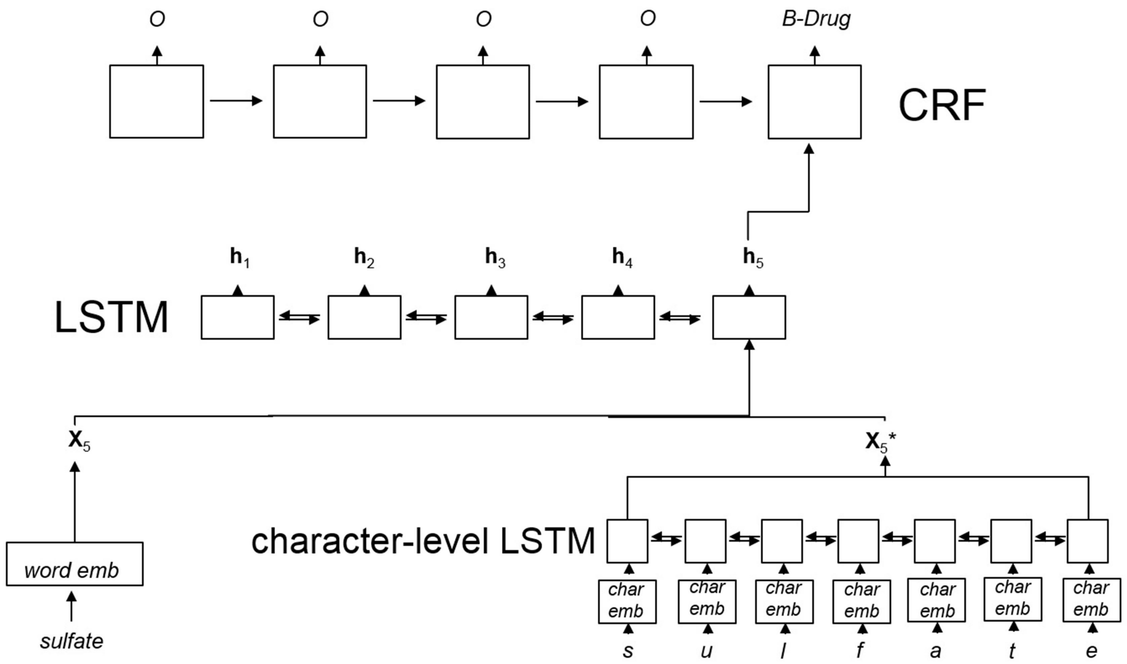


Figure 3.2: The Bidirectional LSTM-CRF with word-level and character-level word embeddings. In the example, word ‘sulfate’ is assumed to be the 5th word in a sentence and its only entity; ‘$\mathbf{x}_5$’ represents its word-level embedding (a single embedding for the whole word); ‘$\mathbf{x}_5^*$’ represents its character-level embedding, formed from the concatenation of the last hidden state of the forward and backward passes of a character-level Bidirectional LSTM; ‘$\mathbf{h}_1$’ - ‘$\mathbf{h}_5$’ are the hidden states of the main Bidirectional LSTM which become the inputs into a final CRF; eventually, the CRF provides the labeling.

3.4 Word features

Neural networks can learn meaningful representations from random initializations of word embeddings. However, it has been proved that pre-trained word embeddings can improve the performance of the network [Lample *et al.*, 2016; Chalapathy *et al.*, 2016a; Dernoncourt *et al.*, 2017; Lee *et al.*, 2016]. In this section, we present the pre-trained embeddings employed in lieu of the random assignments.

3.4.1 Specialized word embeddings

A word embedding maps a word to a numerical vector in a vector space, where semantically-similar words are expected to be assigned similar vectors. To perform this mapping, we have used a well-known algorithm called GloVe [Pennington *et al.*, 2014]. This algorithm learns word embeddings by looking at the co-occurrences of the word in the training data, assuming that a word's meaning is mostly defined by its context and, therefore, words having similar contexts should have similar embeddings. GloVe can be trained from large, general-purpose datasets such as Wikipedia, Gigaword5 or Common Crawl without the need for any manual supervision. In this work, we have experimented with different general purpose, pre-trained word embeddings from the official GloVe website [29] and noticed that the embeddings trained with Common Crawl (cc) (2.2 M unique words) were giving the best results. By default, the code always initializes the word embedding of each unique word in the dictionary with a unique random vector. In alternative, we replace the random initialization with a pre-trained embedding. However, although such datasets generate good embeddings in many cases, for domain-specific tasks such as DNR and CCE they can suffer from some lack of vocabulary. As a matter of fact, in health corpora it is common to find very technical and unusual words which are specific to the health domain. If GloVe is trained only with general-purpose datasets, it is likely that such words will be missing and will still have to be assigned with random vectors.

In order to solve this problem, we have generated a new word embedding by training GloVe from scratch with a large health domain dataset called MIMIC-III [Johnson *et al.*, 2016a]. This dataset contains records of 53,423 distinct hospital admissions of adults to an intensive care unit between 2001 and 2012. The data, structured in 26 tables, include information such as vital signs, observations of care providers, diagnostic codes etc. We expect such a dataset to contain many of the technical words from the health domain that may not appear in general-domain datasets, and as the size of MIMIC-III is sufficiently large, we should be able to extract meaningful vector representations for these words. As a first step, we have selected a subset of the tables and columns, and generated a new dataset where each selected cell together with the title of the corresponding column form a pseudo-sentence. As the next step, we have used this dataset to re-train GloVe, and concatenated these specialized word embeddings with the others to create vectors that contain information from both approaches (cc/mimic). Obviously, there are words that appear in the general dataset, but not in MIMIC-III, and the vice versa. In such cases, the corresponding embedding is still assigned randomly. If a word does not appear in either dataset, we assign its whole embedding randomly. In all cases, the embeddings are updated during training by the backpropagation step.

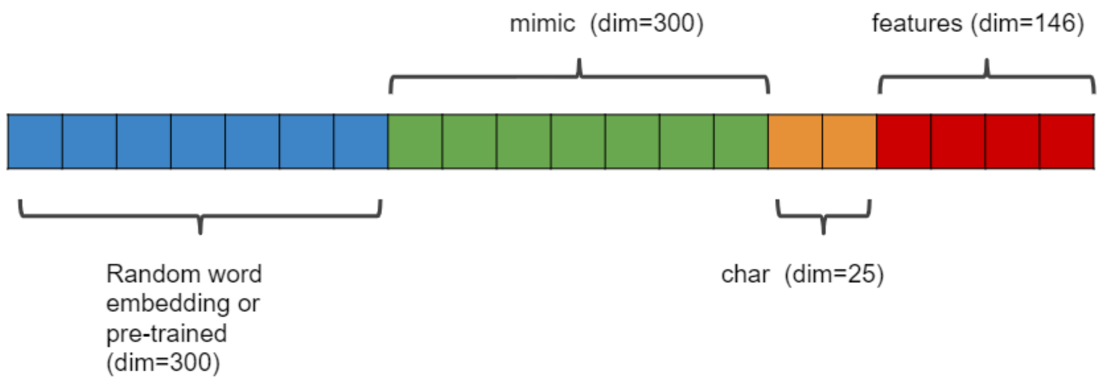


Figure 3.3: Concatenation of all the word features, including general domain embeddings (bleu), specialized embeddings (green), charracter-level embeddings (orange) and handcrafted features (red).

3.4.2 Character-level embeddings

Following [Lampple *et al.*, 2016] we also add character-level embeddings of the words. Such embeddings reflect the actual sequence of characters of a word and have proven to be useful for specific-domain tasks and morphologically-rich languages. Typically, they contribute to catching prefixes and suffixes which are frequent in the domain, and correctly classifying the corresponding words. As an example, a word ending in “cycline”

Feature Types	Features
Morphological	Is all lowercase, is all uppercase, has first letter capitalized, has a letter in the middle capitalized, ends with s, contains digits, is numeric, is alphabetic, is alphanumeric, is a stop word
Semantic	POS tagging, lemma, UMLS concept extracted with MetaMap
Clustering	Index of cluster to which the word embedding belongs. The embeddings clustered (K=20) are the concatenation of Common Crawl and MIMIC-III embeddings (dim=600).

Figure 3.4: Description of the hand-crafted features.

is very likely a drug name, and a character-level embedding could help classify it correctly even if the word was not present in the training vocabulary. All the characters are initialized with a random embedding, and then the embeddings are passed character-by-character to a dedicated LSTM in both forward and backward order. The final outputs in the respective directions promise to be useful encodings of the ending and the beginning of the word. These character-level embeddings are integral part of the LSTM architecture and are not available in the CRF or other models. The character embeddings, too, are updated during training with backpropagation.

3.4.3 Feature augmentation

Conventional machine learning approaches for NER usually have a feature engineering step. [Lee *et al.*, 2016] have shown that adding handcrafted features to a neural network can contribute to increase the recall. In our work, we try this approach with features similar to those used by them. Figure 3.4 shows the list of features used. The distinct values of each feature are encoded onto short random vectors, for a total dimension of 146-D. During training, these encodings are updated as part of the backpropagation step.

3.5 Results

3.5.1 Datasets

Hereafter, we evaluate the models on three datasets in the health domain. In order to ensure that we do not disclose any personally identifiable information and ensure the privacy of any individual that may appear in the documents, we have only used publicly available datasets that have been completely de-identified and approved for research use

	Training	Test
Documents	170	256
Sentences	16315	27626
problem	7073	12592
test	4608	9225
treatment	4844	9344

(a) i2b2/VA

	DDI-DrugBank		*DDI-MedLine*	
	Training	Test	Training	Test
Documents	730	54	175	58
Sentences	6577	145	1627	520
drug_n	124	6	520	115
group	3832	65	234	90
brand	1770	53	36	6
drug	9715	180	1574	171

(b) SemEval-2013 Task 9.1

Table 3.1: Statistics of the training and test datasets used in the experiments

by authoritative ethics bodies.

The first dataset is the *2010 i2b2/VA IRB Revision* (we refer to it as *i2b2/VA* for short in the following) and is used for evaluating CCE. This dataset is a reduced version of the original 2010 i2b2/VA dataset that is no longer distributed due to restrictions introduced by the Institutional Review Board (IRB) in 2011 [Uzuner *et al.*, 2011]. The other two datasets are *DrugBank* and *MedLine*, both part of the *SemEval-2013 Task 9.1* for DNR [Herrero-Zazo *et al.*, 2013]. Table 3.1a and 3.1b describes the basic statistics of these datasets. For the experiments, we have used the official training and test splits released with the distributions.

3.5.2 Evaluation metrics

We report the performance of the model in terms of the F1 score. The F1 score is a very relevant measure as it considers both the precision and the recall, computing a weighted average of them. If we note as TP the number of true positives, FP the false positives and FN the false negatives, we have:

$$\begin{aligned} precision &= \frac{TP}{TP+FP} \\ recall &= \frac{TP}{TP+FN} \\ recall &= \frac{2 \times precision \times recall}{precision + recall} \end{aligned} \tag{3.2}$$

However, it must be remarked that there are different ways of computing the precision and the recall, depending on what we consider as a correct or incorrect prediction [Nadeau and Sekine, 2007]. In this work, following the SemEval-2013 Task 9.1 metrics, we employ the "strict" evaluation method, where both the entity class and its exact boundaries are expected to be correct. We have used the B-I-O tagging standard to annotate the text at word level. In detail, 'B' means the beginning (first word) of a named entity; 'I' stands for 'inside', meaning that the word is part of the same entity (for multi-word entities; e.g., "albuterol sulfate"); and 'O' stands for 'outside', meaning that the word is not part of any named entities. Therefore, a valid annotation of a named entity always begins with a 'B'. An example is shown in Figure 3.5. All the models used in our work have been trained to predict explicit 'B' and 'I' labels for each entity class. The evaluation includes a post-processing step that converts an 'I' prediction to a 'B' if it follows directly an 'O' prediction, thus making all predicted entities valid. An entity is considered as correctly predicted only if all its 'B' and 'I' labels and all its classes are predicted correctly. In the example of Figure 3.5 the prediction will be counted as a true positive only if all the four words "recently diagnosed abdominal carcinomatosis" are tagged as a single entity of the problem class. Every differing 'B' prediction will instead be counted as a false positive. The evaluation protocol explicitly counts only the true positives and the false positives, and derives the false negatives as (number of true entities – true positives).

a)	*Gentleman*	*with*	*recently*	*diagnosed*	*abdominal*	*Carcinomatosis*
	O	*O*	*O*	*O*	*O*	*B-problem*
b)	*Gentleman*	*with*	*recently*	*diagnosed*	*abdominal*	*Carcinomatosis*
	O	*O*	*B-problem*	*I-problem*	*I-problem*	*I-problem*

Figure 3.5: (a) An example of an incorrect tagging in the "strict" evaluation method. (b) An example of a correct tagging in the "strict" evaluation method.

3.5.3 Training and hyper-parameters

For an unbiased evaluation, all the trained models have been tested blindly on unseen test data. In order to facilitate replication of the empirical results, we have used a publicly-available library for the implementation of the neural networks (i.e. the Theano neural network toolkit [Bergstra *et al.*, 2010]) and we release our code[2]. To operate, any machine learning model requires both a set of parameters, which are learned automatically during training, and some "hyper-parameters", which have to be selected manually. Therefore, we have divided the training set of each dataset into two parts: a training set for learning the parameters (70%), and a validation set (30%) for selecting the best hyperparameters [Bergstra and Bengio, 2012]. The hyper-parameters of the LSTM include the number of hidden nodes (for both LSTM versions), $(H_w, H_c) \in \{25, 50, 100\}$; the word embedding dimension, $d_w \in \{50, 100, 200, 300, 600\}$; and the character embedding dimension, $d_c \in \{25, 50, 100\}$. Additional hyper-parameters include the learning rate and the drop-out rate, which were left to their default values of [0.01] and [0.5] respectively [Srivastava *et al.*, 2014]. All weights in the network, feature encodings and the words that do not have a pre-trained word embedding have been initialized randomly from the uniform distribution within range [1,1], and updated during training with backpropagation. The number of training epochs was set to 100, selecting the epoch that obtained the best results on the validation set. The best model from the validation set was finally tested on the unseen, independent test set without any further tuning, and the corresponding accuracy reported in the tables. Table 3.2 shows all the hyper-parameters used for the experiments reported in the Section 3.5.4.

[2]HealthNER. Available from: https://github.com/ijauregiCMCRC/healthNER

Hyper-parameter	Value
Word embedding dim (d_w)	300(cc)/600(cc/mimic)
Word LSTM hidden layer dim (H_w)	100
Char embedding dim (d_c)	25
Char LSTM hidden layer dim (H_c)	25
Dropout	0.5
Optimization Stochastic Gradient Descend Learning rate	0.01
Concatenated hand-crafted features dim	146

Table 3.2: The hyper-parameters used in the final experiments

3.5.4 Results

Table 3.3a and 3.3b shows the results of the proposed models and the state-of-the-art systems on the CCE task (*i2b2/VA* dataset) and DNR task (*DrugBank* and *MedLine* datasets), respectively. In the following subsections, we discuss the results obtained for each task.

3.5.4.1 CCE results over the i2b2/VA dataset

On the *i2b2/VA* dataset (Table 3.3a), the Bidirectional LSTM-CRF (BLSTM-CRF) with Common Crawl embeddings (cc) and character-level embeddings (char) as features has obtained the best results (83.35% F1 score). The model has outperformed all systems from the literature (top quadrant of Table 3.3a) which are all based on conventional domain-specific feature engineering. It is important to note that [de Bruijn *et al.*, 2011] had reported a higher accuracy on 2010 i2b2/VA (85.23% F1 score), but their model was trained and tested on the original version of the dataset which is no longer available due to the restrictions introduced by the Institutional Review Board. As for what specialized embeddings are concerned, Table 3.4 shows that the general-domain dataset Common Crawl already contains almost all the words in the dataset. Therefore, adding the MIMIC-III embeddings (cc/mimic) does not extend the vocabulary, and therefore it brings no improvement. On the other hand, the B-LSTM has improved by 0.3 pp with the cc/mimic embeddings. Even though the mimic embeddings do not cover significant extra vocabulary, they may have enriched the feature space. Conversely, the cc/mimic embeddings have provided no improvements with the B-LSTM-CRF. For this, we need to take into account that the BLSTM-CRF already has a high score (83.35% F1-score). Consequently, it may be more difficult to improve its results. Conversely, using conventional feature engineering has led to lower accuracy (77.81% F1-score). Eventually, concatenating both the features and the pre-trained embeddings showed no improvement over the best model. Table 3.3a also shows the importance of using a final CRF layer in the B-LSTM-CRF, given that the B-LSTM alone was only able to achieve a 77.59% F1 score. At its turn, the CRF baseline has only obtained a 64.09% F1 score in its best configuration, lower than any version of the LSTM.

Model	F1-score (%)
Binarized Neural Embedding CRF [Wu *et al.*, 2015]	82.80
CliNER [Boag *et al.*, 2015]	80.00
Truecasing CRFSuite [Fu and Ananiadou, 2014]	75.86
CRF + (random)	11.27
CRF + (features)	25.53
CRF + (cc)	53.72
CRF + (cc/mimic)	58.28
CRF + (cc/mimic) + (features)	64.09
B-LSTM + (random)	65.43
B-LSTM + (random) + (features)	69.42
B-LSTM + (cc)	75.17
B-LSTM + (cc) + (char)	76.79
B-LSTM + (cc/mimic) + (char)	77.19
B-LSTM + (cc/mimic) + (char) + (features)	77.59
B-LSTM-CRF + (random)	75.05
B-LSTM-CRF + (random) + (features)	77.81
B-LSTM-CRF + (cc)	82.85
B-LSTM-CRF + (cc) + (char)	**83.35**
B-LSTM-CRF + (cc/mimic) + (char)	82.70
B-LSTM-CRF + (cc/mimic) + (char) + (features)	83.29

(a) CCE results over the *i2b2/VA* dataset

Model	*DDI-DB* F1(%)	*DDI-ML* F1(%)
WBI-NER [Rocktäschel *et al.*, 2013]	87.80	58.10
Hybrid-DDI [Abacha *et al.*, 2015]	80.00	37.00
Word2Vec+DINTO [Segura-Bedmar *et al.*, 2015]	75.00	57.00
CRF + (random)	28.70	13.65
CRF + (features)	44.52	20.19
CRF + (cc)	43.42	32.62
CRF + (cc/mimic)	53.12	30.87
CRF + (cc/mimic) + (features)	66.45	29.36
B-LSTM + (random)	65.09	21.28
B-LSTM + (random) + (features)	75.43	30.88
B-LSTM + (cc)	71.75	42.39
B-LSTM + (cc) + (char)	84.35	43.33
B-LSTM + (cc/mimic) + (char)	83.63	44.39
B-LSTM + (cc/mimic) + (char) + (features)	84.06	45.92
B-LSTM-CRF + (random)	69.50	44.60
B-LSTM-CRF + (random) + (features)	75.78	43.36
B-LSTM-CRF + (cc)	79.03	57.87
B-LSTM-CRF + (cc) + (char)	87.87	59.02
B-LSTM-CRF + (cc/mimic) + (char)	**88.38**	**60.66**
B-LSTM-CRF + (cc/mimic) + (char) + (features)	87.42	59.75

(b) DNR results over the *DrugBank* and *MedLine* datasets

Table 3.3: Comparison of the results between the different RNN models and the state-of-the-art systems over the CCE and DNR tasks.

3.5.4.2 DNR results over the DrugBank and MedLine datasets

In the DNR task (Table 3.3b), the proposed B-LSTM-CRF with the concatenated word embeddings (cc/mimic) and the character-level embeddings (char) has improved over all the previous approaches on both *DrugBank* (88.38% F1 score) and *MedLine* (60.66% F1 score). Table 3.4 shows that only 49% of the words in the datasets have been found in the cc embeddings. However, when the concatenated embeddings (cc/mimic) are used, the percentage of found words has increased to 67% for *DrugBank* and 61% for *MedLine*, leading to better results in the classification task. Words that appear in the MIMIC-III dataset but are not contained in Common Crawl are typically very technical and domain-specific, such as drug names or treatments; examples include: *pentostatin*, *sitagliptin*, *hydrobromide*, *organophosphate*, *pyhisiological* and *methimazole*. In total, 1189 extra words have been mapped in *DrugBank* and 716 in *MedLine* thanks to the use of MIMIC-III. However, the B-LSTM has only obtained an accuracy improvement on the *MedLine* dataset, but not on *DrugBank*. This can be explained by the fact that the accuracy of the B-LSTM on *MedLine* is very low (44.33%) and, therefore, easy to improve. Instead, on *DrugBank* the accuracy of the B-LSTM is already very high (84.35% F1-score) and thus difficult to improve. With the B-LSTM-CRF, results with extra vocabulary covered by the cc/mimic embeddings have improved with both datasets.

As for what concerns the hand-crafted features, their use has led to higher accuracy than with the Common Crawl embeddings on the *DrugBank* dataset in two cases. However, the concatenation of the features and the pre-trained embeddings has not improved the best results. As in the CCE task, the B-LSTM-CRF model has proved better than the B-LSTM alone on both *DrugBank* (88.38% vs 84.35% F1-score) and *MedLine* (60.66% vs 45.92% F1-score.) Finally, we can see that the use of the character-level embeddings has led to higher relative improvements for *DrugBank* than for the other two datasets. A plausible explanation for this is that this dataset contains more words with distinctive prefixes and suffixes which are more effectively captured by the character-level embeddings.

In general, the CRF has significantly underperformed compared to the neural networks. We speculate that this model may require more extensive feature engineering to achieve a comparable performance, or that it may not be able to achieve it at all. In particular, we see that the CRF has performed the worst with *MedLine*. A possible explanation

	Common Crawl (cc)	*Common Crawl + MIMIC-III* (cc+mimic)
i2b2/VA	99.99 %	99.99 %
DDI-DrugBank	49.50 %	67.02 %
DDI-MedLine	49.10 %	61.51 %

Table 3.4: Percentage of words initialized with pre-trained embeddings in the train, dev and test of the respective datasets.

can be found in the "curse of dimensionality": *MedLine* is a small dataset (1627 training sentences), while the overall dimensionality of the input embeddings is 746. This makes the learning problem very sparse and seems to seriously affect a linear model such as the CRF. On the contrary, the non-linear internal architecture of the neural networks may in some cases help reduce the effective dimensionality and mollify this problem.

3.5.4.3 Accuracy by entity classes

Table 3.5a and 3.5b break down the results by entity class for the best model on each dataset. With the *MedLine* dataset, we can notice the poor performance at detecting *brand*. In *DrugBank*, the same issue occurs with entity class *drug_n*. This issue is likely attributable to the small sample size. Instead, the *i2b2/VA* dataset all entity classes are detected with similar F1 scores, likely owing to the larger number of samples per class. However, we see that *brand* achieves the second best F1 score in *DrugBank* despite its relatively low frequency in the dataset, and that *drug_n* obtains a very poor performance in *MedLine* even if it has the second highest frequency. We identify two other main factors that may have a major impact on the accuracy: (1) the average length of the entities in each class, and (2) the number of test entities that had not been seen during the training stage. In this respect, the *brand* and *drug* entities are usually very short (average $\sim$ 1 word), while the *group* and *drug_n* entities often have multiple words. Since shorter entities are easier to predict correctly, *brand* obtains better accuracy than group in *DrugBank*. On the other hand, the *drug_n* and *group* entities have similar length, but in *MedLine drug_n* obtains a very poor performance. This is most likely because no entity of type *drug_n* that appears in the test set had been seen during training. Conversely, a large percentage of the test *group* entities had been seen during training and have therefore proved easier

	Entity	Precision	Recall	F1-score
B-LSTM-CRF+(cc)+(char)	problem	81.29	83.62	82.44
	test	84.74	85.01	84.87
	treatment	83.36	83.55	83.46

(a) *i2b2/VA*

	Entity	*DDI-DrugBank*			*DDI-MedLine*		
		Precision	Recall	F1-score	Precision	Recall	F1-score
B-LSTM-CRF	group	81.69	87.88	84.67	69.14	60.22	64.37
	drug	94.77	89.56	91.83	73.89	77.33	75.57
+(cc+mimic)	brand	84.21	90.57	87.27	100.00	16.67	28.57
+(char)	drug_n	00.00	00.00	00.00	68.18	25.57	37.19

(b) *SemEval-2013 Task 9.1*

Table 3.5: Results by class for the B-LSTM-CRF with character-level and cc/mimic embeddings.

to predict.

3.6 Conclusion

We have set to investigate the effectiveness of the Bidirectional LSTM and Bidirectional LSTM-CRF – two specific architectures of recurrent neural networks – for drug name recognition and clinical concept extraction, and compared them with a baseline CRF model. As input features, we have applied combinations of different word embeddings (Common Crawl and MIMIC–III), character-level embedding and conventional feature engineering. We have showed that the neural network models have obtained significantly better results than the CRF, and reported state-of-the-art results over the i2b2/VA, DrugBank and MedLine datasets using the B-LSTM-CRF model. We have also provided evidence that retraining GloVe on a domain-specific dataset such as MIMIC-III can help learn vector representations for domain-specific words and increase the classification accuracy. Finally, we have showed that adding hand-crafted features does not further improve performance since the neural networks can learn useful word representations automatically from pre-trained word embeddings. Consequently, time-consuming, domain-specific feature engineering can be usefully avoided.

Chapter 4

English-Basque Statistical and Neural Machine Translation

4.1 Introduction

Machine Translation (MT) is one of the oldest tasks in NLP, dating back to the 1950s, yet it is a partially unresolved challenge to the present days. The goal of MT is to read and understand a sentence in a *source* language to the extent required to generate its correct translation in a *target language*. The advent of deep neural networks has also led to significant progress in MT, now popularly known as Neural Machine Translation (NMT) [Sutskever *et al.*, 2014; Bahdanau *et al.*, 2015; Luong *et al.*, 2015; Vaswani *et al.*, 2017; Wu *et al.*, 2016]. NMT models mainly use the encoder-decoder architecture (described in Section 2.3.4.2) and have outperformed previous translation systems in many language pairs (e.g., German-English, French-English).

However, in order to reach high accuracies, neural translation systems tend to require very large parallel training corpora [Koehn and Knowles, 2017]. As a matter of fact, such corpora are not yet available for many language pairs. When the training data are relatively small, other, more traditional approaches such as Statistical Machine Translation (SMT) [Koehn *et al.*, 2007] seem to tend to be more accurate. Various researchers have compared the quality of PB-SMT and NMT translation models [Bentivogli *et al.*, 2016; Toral and Sánchez-Cartagena, 2017; Castilho *et al.*, 2017] on different language pairs and with different training corpus sizes. In recent years, several ideas have been proposed in order to mollify the issue of not having enough training data. This includes using multi-lingual systems with zero-shot translation [Johnson *et al.*, 2016b], transfer learning [Zoph *et al.*, 2016] and back-translations [Sennrich *et al.*, 2016]. Several work has also been proposed in the popular WMT workshops with a variaety of low resouced language pairs. Zhou et al. [2018] have explored effective methods for cross-lingual transfer learning to translate from rich-resource languages to low-resource languages. Their research has shown that using neighbouring language families for the cross-lingual transfer often can achieve better performance than a single-family multi-source multi-target baseline. They have reported a 9.9 BLEU p.p. improvement for English-Swedish translation. Similarly, Kocmi and Bojar [2018] show that a trivial transfer learning, where they first train a "parent" model for a high-resource language pair and then continue the training on a low-resource pair only by replacing the training corpus, can bring significant translation performance improvement even in distant unrelated languages. Using parallel synthetic data, by leveraging monolingual data with back-translation or other approaches has also been a popular proposal in the workshops [Chinea-Rios *et al.*, 2017; Currey *et al.*, 2017]. Finally, meta-learning [Gu *et al.*, 2018] has been also proposed as an interesting training method for low resourced translation models.

For this work, we have selected a low-resource language pair, English-Basque (abbreviation: en-eu), since I am a Basque native speaker and this language has had limited dedicated studies, and used it as a case study for a comparison of statistical and neural machine translation. Past machine translators for the Basque language have mainly used rule-based [Mayor *et al.*, 2011] and statistical approaches [de Ilarraza *et al.*, 2008; Stroppa *et al.*, 2006]. In the WMT 2016 workshop [Bojar *et al.*, 2016] an IT-domain translation shared task was organized which included English-Basque as a language pair. Del Gaudio et al. [2016], participants on the shared-task, have proposed two alternative systems, one based on PB-SMT [Koehn *et al.*, 2007] and a system exploiting deep language engineering approaches [Žabokrtskỳ *et al.*, 2008]. Their work showed how challenging it is to translate in this distant language pair, achieving translations with low accuracy with respect to the reference sentences ($\sim$ 10 BLEU). In our work, we compare three different systems: OpenNMT, an open-source NMT system; Moses SMT, an open-source SMT system; Apertium; a publicly available rule-based machine translation (RBMT) system and Google Translate, a publicly-available commercial system which uses either SMT or NMT models depending on the language pair. The first two have been trained by us with dedicated datasets, while Google has just been used "as is" from its API. The three models have been tested over open-domain and Information Technology (IT) domain datasets from the WMT2016 IT helpdesk shared task. Moreover, we have released a new, small en-eu corpus (named *Berriak*) useful for probing English-Basque machine translation. This corpus consists of 500 long and complex sentences translated from English to Basque by experienced translators and is much more realistic and challenging than the existing en-eu corpora. Due its small size, we have only used it as a test set.

4.2 The Basque Language

Basque (*Euskara*) is a language spoken in the Basque Country (northern Spain and south-western France). It is not considered a member of the Indo-European language family and it remains isolated to date, meaning that researchers have not found any other language with similar characteristics. As stated by [Mayor *et al.*, 2011], Basque is an *agglutinative* language with rich inflectional morphology. This means that a word may include several morphemes that change its inflectional category such as number, case, tense or person. Unlike in fusional languages such as Spanish and French, in agglutinative languages the boundaries between morphemes remain clear-cut [Aikhenvald, 2007]. For agglutinative languages, rule-based systems have proved a rather effective translation approach in the past [Koehn and Monz, 2006].

During the military dictatorship of Spain (1939-1975), Basque became illegal and the number of speakers dropped drastically. However, in the 1970s a process for the formal standardization of the language (*Euskara Batua*) began and Basque teaching in schools was restored. Nowadays, the language is official in the Basque autonomous community in Spain and it has reached approximately 1M speakers (including Navarre and the French side). Yet, the persisting lack of available translation corpora makes the development of Basque machine translators a difficult task.

4.3 Methods

4.3.1 Moses SMT

SMT has been the state-of-the-art approach to machine translation for many years [Koehn *et al.*, 2003]. SMT systems are usually phrase-based system which first try to learn the alignments of phrases between different languages, and then predict the best composition of phrases for the translation with the help of a target language model (LM).

In this work, we have evaluated a popular open-source SMT toolkit called Moses [Koehn *et al.*, 2007]. First, a word alignment model between the source and target languages has been learned over the training data with the GIZA++ toolkit [Och and Ney, 2003]. Then, an LM has been learned over the training data with KenLM [Heafield, 2011]. Finally, based on these two models, the Moses decoder has been used to translate the sentences. The parameters of the models have been kept to the default recommended values by the authors of Moses.

4.3.2 Apertium

Apertium[1] is a RBMT system that is only available to translate from Basque to English (and not the other way around). It combines multiple morphological, lexical, chunking and anaphora resolution rules.

4.3.3 Google Translate

Google Translate is a publicly-available, well-known commercial machine translator. Recently, it has implemented the Google Neural Machine Translation (GNMT) [Johnson *et al.*, 2016b] over many language pairs and en-eu is one of them[2]. Google Translate does not train from a dedicated annotation of parallel text; rather, it crawls the Web to forage for "likely parallel" paragraphs (for instance, those marked with multiple HTML `lang` tags). In our work, we have used it from the convenient Google Cloud Translation API[3].

[1] `https://apertium.org/index.eng.html?dir=eus-eng#translation`
[2] https://cloud.google.com/translate/docs/languages
[3] Google Translate API: https://cloud.google.com/translate/docs/

4.3.4 OpenNMT

We have trained an NMT model by using the OpenNMT toolkit [Klein *et al.*, 2017] with the *seq2seq* architecture of [Sutskever *et al.*, 2014]. This architecture is formed by an encoder, which converts the source sentence into a sequence of numerical vectors, and a decoder, which predicts the target sentence based on the encoded source sentence. Both the encoder and the decoder are usually recurrent neural networks (RNNs). Additionally, an *attention mechanism* [Bahdanau *et al.*, 2015; Luong *et al.*, 2015] has been used to learn soft-alignments between the source and the target sentences.

In our model, we have used the Long Short-Term Memory (LSTM) network [Hochreiter and Schmidhuber, 1997] for both the encoder and the decoder. We have also used the *dot* attention mechanism [Luong *et al.*, 2015] where weights over the encoded source sentence are provided by an auxiliary network. Like with any other neural models for NLP, prior to processing each unique word in the corpus needs to be mapped to a high-dimensional vector (*word embedding*). This mapping can be either random (the default) or based on user-provided pre-trained embeddings. In addition, the word embeddings can be kept constant during training, or updated alongside all other parameters to minimise the cost function. Since pre-trained embeddings have typically reported higher accuracies [Dernoncourt *et al.*, 2017; Lample *et al.*, 2016], we have trained Basque word embeddings using GloVe [Pennington *et al.*, 2014] over the Basque Wikipedia. For English, we have used the available CommonCrawl pre-trained embeddings[4]. We have evaluated the use of these embeddings in two different ways: maintaining them fixed during both training and testing (f-emb) and updating them during the training stage (u-emb). The word embeddings have a dimension of 300. The remaining parameters of the network have been kept to their default recommended values by the authors of OpenNMT.

[4]GloVe: https://nlp.stanford.edu/projects/glove/

Data	*PaCo2_EnEu*	*WMT16_IT*	*Berriak*
training	125,356	89,983	—
test	5,000	1,000	500

Table 4.1: The number of samples in the *PaCo_EnEu*, *WMT16_IT* and *Berriak* datasets.

4.4 Experiments

4.4.1 Corpora

As mentioned previously, the en-eu language pair is considered to be low-resourced. To mitigate this issue, the WMT16 machine translation for IT domain shared task[5] provided a parallel en-eu corpus. This corpus includes both an IT-domain dataset and an open-domain dataset (*PaCo2_EnEu*). *PaCo2_EnEu* consists of approximately 130,000 en-eu translations crawled from the web [San Vicente *et al.*, 2012]. In the experiments, we have used 5,000 as a test set and the rest for training the models (in the shared-task *PaCo2_EnEu* was only used as out-of-domain training data, not as testing data). We have also used the IT-domain data to evaluate the translators over a specialised domain. The IT-domain training set consists of 89,983 samples, but only 2,000 of them are proper sentences; the rest are translations of IT terms from Wikipedia and localization PO files. Consequently, the amount of "good quality" data in the training set is very limited. At its turn, the test set consists of 1,000 proper sentences. Another English-Basque parallel corpus is available at the OPUS repository [6], which is larger in number of sentences ($\sim$ 1M). However, the sentence alignment in this corpus is not good [Tiedemann and Scherrer, 2017], resulting in a rather noisy dataset, and that can affect the quality of the predictions made by the MT system [Khayrallah and Koehn, 2018]. This corpus is more suited for training document-level NMT models [Miculicich *et al.*, 2018], because it has clear document boundaries, and because the surrounding sentence can help to understand the context better, achieving better translations and mollifying the issues produced by poor sentence alignment. Therefore, we have used this dataset in Chapter 6 to train our document-level translation model.

By inspecting these resources, we had realised the lack of long and complex sen-

[5]WMT16: `http://www.statmt.org/wmt16/it-translation-task.html`
[6]OPUS: `http://opus.nlpl.eu/`

tences, likely a major limitation for the realistic evaluation of this language pair. Such sentences appear in most professional translations and they are expected to prove far more challenging for automated translators. To provide a resource contribution, we have therefore collected and released a small, high-quality en-eu corpus called *Berriak* (*news* in Basque)[7],[8]. To create a suitable corpus, we have randomly selected English sentences from the English-German news corpus of WMT16 which meets the requirements. The sentences have been translated into Basque with the help of Librezale[9], an open group of highly-qualified volunteers who work to increase the use of the Basque language in the IT domain. To date, we have collected 500 en-eu translations for a total of 10,280 tokens. The number of distinct tokens in the corpus is 4,554 for Basque and 3,741 for English, and the average sentence length is $\sim$19 and $\sim$23 tokens respectively. Usually morphologically rich languages such as Basque tend to have a larger vocabulary size. Due to the low number of sentences in this corpus, in this work we have only used it as a further test set for models trained with *PaCo2_EnEu*. The corpus is similar to the popular news domain translations released by the organizers of the WMT workshops every year, and we believe will allow for a better comparison between translation models and to expose brittleness in models trained on existing benchmarks [Liu *et al.*, 2019]. The manual translations are still ongoing and we intend to release an extended version in the near future. Table 4.1 summarises the number of samples of the various datasets.

4.4.2 Experimental Settings and Results

We have conducted a number of experiments to evaluate the models in a variety of scenarios. In the first experiment, we have trained the SMT and NMT models with the *PaCo2_EnEu* training set and tested them with both the *PaCo2_EnEu* test set and *Berriak*. Google Translate has been used as is. Experiments have been conducted in both English-to-Basque (en$\rightarrow$eu) and Basque-to-English (eu$\rightarrow$en) to assess performance in both directions. In addition, for the NMT model we have experimented with updated random embeddings (r-emb), fixed pre-trained embeddings (f-emb), and updated pre-trained embeddings (u-emb). Table 4.2 reports the BLEU scores [Papineni *et al.*, 2002] for the three

[7]ISLRN: 197-383-395-000-1
[8]https://github.com/ijauregiCMCRC/english_basque_MT
[9]Librezale: https://librezale.eus/

Model	***PaCo2_EnEu***	***Berriak***
en→eu		
Moses SMT	21.02	5.90
OpenNMT(r-emb)	20.07	1.49
OpenNMT(f-emb)	19.39	1.43
OpenNMT(u-emb)	21.18	1.84
Google Translate	9.12	9.91
eu→en		
Moses SMT	24.20	8.53
OpenNMT(r-emb)	19.44	3.35
OpenNMT(f-emb)	18.61	4.28
OpenNMT(u-emb)	22.42	5.53
Google Translate	16.66	20.80
Apertium	8.15	7.3

Table 4.2: BLEU score of the models over the *PaCo_EnEu* and *Berriak* corpora.

models. The first remark is that all models generally perform worse with Basque as the target language. This is in line with the literature, in relation to the fact that morphologically rich languages are harder as target, comparing to other languages with simpler morphology such as English [Bojar *et al.*, 2013]. The main issue is that treating words as atomic units of information in morphologically rich languages is not suitable [Passban, 2017], as morphemes can significantly change the meaning of the word. As future work, better morphological decomposition or subword tokenization [Sennrich *et al.*, 2015] could be explored. As for the models' comparison, both Moses SMT and OpenNMT have remarkably outperformed Google Translate on the *PaCo2_EnEu* test set. Apertium, the RBMT system, has significantly underperformed compared to the other systems. The NMT model has achieved the highest BLEU score (21.18) in the en→eu direction, while the SMT model has achieved the highest BLEU score (24.20) in the opposite direction. This also is in line with recent work on morphologically rich languages [Belinkov *et al.*, 2020]. For the NMT model, updating the pre-trained embeddings during training (u-emb) has invariably led to the highest accuracies, up to an improvement of 2.98 BLEU points over the random embeddings in the eu→en direction.

However, the performance ranking has changed drastically when testing on the more probing *Berriak* corpus. In this case, Google Translate has achieved the highest BLEU scores by a large extent. We believe that both Moses SMT and OpenNMT, which have been trained using only the *PaCo2_EnEu* training set, have obtained such low results because the training corpus does not contain the same kind of long sentences as *Berriak*

Model	***WMT16_IT***
en→eu	
Moses SMT	11.74
Moses SMT+(PaCo train)	11.89
OpenNMT(r-emb)	11.87
OpenNMT(PaCo train)(r-emb)	12.42
OpenNMT(u-emb)	12.75
OpenNMT(PaCo train)(u-emb)	12.31
Google Translate	14.46
eu→en	
Moses SMT	19.06
Moses SMT+(PaCo train)	19.34
OpenNMT(r-emb)	15.46
OpenNMT(PaCo train)(r-emb)	16.93
OpenNMT(u-emb)	17.30
OpenNMT(PaCo train)(u-emb)	18.01
Google Translate	24.66
Apertium	6.6

Table 4.3: BLEU score of the models over the *WMT16_IT* corpus.

and therefore the models could not learn to translate such challenging sentences. Between SMT and NMT, the former has clearly outperformed the latter, confirming that SMT generalises better when the training corpus is limited. In this case, Apertium (7.3 BLEU) has also performed better than the NMT model. On the other hand, the training corpus of Google Translate is certainly much bigger, and that has helped it achieve better results on *Berriak*. However, the BLEU score when Basque is the target is still very low (9.91) and significant improvements are an outstanding need. For what concerns NMT and word embedding, also in this case the updated pre-trained embeddings have led to an improvement (although slight) in score.

In a second experiment in the IT domain (Table 4.3), Google Translate has again obtained the best results, and the RBMT model has significantly underperformed. This can be explained with the fact that Moses SMT and OpenNMT have only been trained with 2,000 proper IT-domain sentences. Between these two models, OpenNMT has outperformed Moses SMT for Basque as the target language, and vice versa for English. To mollify the small training size issue, we have added the open-domain corpus to the training data (noted as *PaCo train* in Table 4.3). The results have slightly improved for both NMT and SMT, with a more noticeable improvement for NMT (12.42 in en→eu and 16.93 in eu→en). Larger relative improvements have been achieved with the use of the pre-trained embeddings (12.75 in en→eu and 17.30 in eu→en). Since the updated embeddings had proved clearly more accurate in the previous experiment, we have not used the fixed embeddings in this experiment. Finally, using both the open-domain data and the pre-trained embeddings has only improved the scores for English as the target language. Once again, all the models have performed significantly better with English as the target language, with an even bigger margin compared to the general-domain experiment. We speculate that this may be due to the fact that in the IT domain the Basque language does not have a vocabulary as comprehensive and developed as English does. In fact, many IT words and expressions are taken from English unchanged.

For a qualitative analysis, Table 4.5 shows three examples of translations provided by the different models alongside the ground truth from the *PaCo2_EnEu* test set, which is the dataset on which the trained NMT and SMT models have obtained the best accuracies. We can see that for sentence 2 the OpenNMT model has provided a translation identical to the ground truth, probably thanks to the fact that there are sentences with the same structure in the training corpus. However, the NMT model tends to directly bypass many words from the original English sentence into the prediction (see sentences 1 and 3; NB: OpenNMT allows the model to bypass words from the source sentence). More precisely, if the model is uncertain about which word from the target vocabulary should be predicted next, it will pass on the word with highest attention weight from the source sentence. This mechanism aims to help the prediction of words such as proper names, which are not likely to appear in the target vocabulary. As additional analisys, we have computed the percentage of words bypassed by the different NMT models. To this aim, we have counted the number of words in the test set's predictions which did not belong to the target vocabulary, and divided it by the total number of words predicted. Table 4.4 shows the computed percentages, averaged over all the different NMT models. We can observe that the numbers are clearly higher when Basque is the target language, which matches our intuition that Basque is more difficult to translate into. When comparing the different datasets, we observe a trend to bypass more words in *Berriak*, which is understandable as this dataset does not have a training corpus and has longer and more complex sentences. Conversely, *WMT16_IT* has the lowest percentages of source words in the predictions. This is likely due to the fact that this dataset is very domain-specific and has a smaller

Corpus	Bypassed (%)
en→eu	
PaCo2_EnEu	21.70
Berriak	36.60
WMT16_IT	3.78
eu→en	
PaCo2_EnEu	4.29
Berriak	7.59
WMT16_IT	1.07

Table 4.4: Average of the percentages of bypassed words by all the NMT models in each dataset and each direction.

vocabulary size.

On a separate note, NMT tends to predict the same word repeatedly (see sentence 3), as often been reported for neural encoder-decoder architectures [Ding *et al.*, 2017]. On the other hand, the SMT model seems able to match more words correctly in each sentence, but it has difficulties to form grammatically-complete sentences (see all three examples). Finally, the sentences predicted by Google Translate contain synonyms of the words in the ground truth (e.g., *Iruñean* vs *Pamplona*) and errors in the inflectional morphemes (e.g., *entzierroa* vs *entzierroetan*, *Bilaketa* vs *Bilaketaren*, *zezenketen* vs *zezenketarako*).

4.5 Conclusion

In this work we have presented a performance comparison of three contemporary MT approaches on a low-resourced language pair, English-Basque. The compared approaches include an NMT model (OpenNMT, with and LSTM encoder/decoder), an SMT model (Moses) and the popular Google Translate service.

The experimental results have showed that all the models have achieved worse results when Basque is the target language, confirming that languages with rich morphology are more difficult to translate into. The NMT and SMT models have outperformed Google Translate when using training and test data from the same corpus (*PaCo2_EnEu*). However, these models have not generalised well on long and complex sentences, in contrast to Google Translate. For the NMT model, initialising the word embeddings with pre-trained embeddings (based on the Basque Wikipedia for Basque and CommonCrawl for English) and updating them during training has invariably led to the best BLEU scores.

Thanks to the embedding initialization, the NMT model has been able to improve the results obtained by the SMT. In absolute terms, the achieved BLEU scores suggest that machine translation for Basque still has large margins for improvement, mainly due to the lack of available billingual corpora for this language pair. In order to contribute in the alleviation of this problem, we have released a new, small corpus (named *Berriak*) of highly accurate en-eu sentences translated by experienced human translators to be used as a probing test set for this language pair. However, there is a need to propose alternative RNN architectures that will handle better the lack of data.

English sentence	1. *How many people take part in The Sanfermin Bullrunnings - Sanfermin.com - Pamplona*
	2. *Search results as from 07/02/2011 in "Zarzuela"*
	3. *For this reason, after Madrid and Sydney, they plan to continue this international anti-bullfighting campaign in Croatia and Berlin.*
Ground Truth	1. *Sanferminetako entzierroa zenbat jendek egiten duen - Sanfermin.com - Pamplona*
	2. *Bilaketa emaitzak 2011/07/02 egunetik aurrera “Zarzuela”-(e)n*
	3. *Horregatik, zezenketen eta entzierroen aurkako nazioarteko protesta Madrilen eta Sidneyn egiteaz gain, Kroazia eta Berlinen ere izango da.*
Moses SMT	1. *Zenbat jende parte hartzeko Sanferminetako Bullrunnings - Sanfermin.com - Pamplona*
	2. *Bilaketa emaitzak 2011/07/02 egunetik aurrera “Zarzuela”-(e)*
	3. Hori dela eta, ondoren, Madrilera eta Sydney jarraitzen dute, plan horrek nazioarteko aurkako Zezenketetako @-@ kanpaina batean Kroazia eta Berlingo.
OpenNMT	1. *Nola bizi da Sanfermin Bullrunnings - Sanfermin.com - Pamplona*
	2. *Bilaketa emaitzak 2011/07/02 egunetik aurrera “Zarzuela”-(e)n*
	3. *Hori dela eta, Madrilen, Madrilen, Madrid, bullfighting eta Berlin, international eta Berlin.*
Google	1. *Zenbat pertsona parte hartu Sanferminetako entzierroetan - Sanfermin.com - Iruñean*
	2. *Bilaketaren emaitzak 2011/02/07 “Zarzuela” -en*
	3. *Horregatik, Madrilen eta Sydneyen ondoren, Kroazia eta Berlinen kontrako zezenketarako nazioarteko kanpaina aurrera eramateko asmoa dute.*

Table 4.5: Example of translations over the *PaCo2_EnEU* (en→eu) test set.

Chapter 5

Regressing Word and Sentence Embeddings for Regularization of Neural Machine Translation

5.1 Introduction

Machine translation (MT) is a field of natural language processing (NLP) focussing on the automatic translation of sentences from a *source* language to a *target* language. In recent years, the field has been progressing quickly mainly thanks to the advances in deep learning and the advent of neural machine translation (NMT). The first NMT model was presented in 2014 by Sutskever et al. [Sutskever *et al.*, 2014] and consisted of a plain encoder-decoder architecture based on recurrent neural networks (RNNs). In the following years, a series of improvements has led to major performance increases, including the attention mechanism (a word-aligment model between words in the source and target sentences) [Bahdanau *et al.*, 2015; Luong *et al.*, 2015] and the transformer (a non-recurrent neural network that offers an alternative to RNNs and makes NMT highly parallelizable) [Vaswani *et al.*, 2017]. As a result, NMT models have rapidly outperformed traditional approaches such as phrase-based statistical machine translation (PBSMT) [Koehn *et al.*, 2007] in challenging translation contexts (e.g., the WMT conference series). Nowadays, the majority of commercial MT systems utilise NMT in some form.

However, NMT systems are not exempt from limitations. The main is their tendence to overfit the training set due to their large number of parameters. This issue is common to many other tasks that use deep learning models and it is caused to a large extent by the way these models are trained: maximum likelihood estimation (MLE). As pointed out by Elbayad et al. [Elbayad *et al.*, 2018], in the case of machine translation, MLE has two clear shortcomings that contribute to overfitting:

1. **Single ground-truth reference**: Usually, NMT models are trained with translation examples that have a single reference translation in the target language. MLE tries to give all the probability to the words of the ground-truth reference and zero to all others. Nevertheless, a translation that uses different words from the reference (e.g. paraphrase sentences, synonyms) can be equally correct. Standard MLE training is not able to leverage this type of information since it treats every word other than the ground truth as completely incorrect.

2. **Exposure bias**[Bengio *et al.*, 2015]: NMT models are trained with "teacher forcing", which means that the previous word from the reference sentence is given as input to the decoder for the prediction of the next. This is done to speed up training convergence and avoid prediction drift. However, at test time, due to the fact that the reference is not available, the model has to rely on its own predictions and the performance can be drastically lower.

Both these limitations can be mitigated with sufficient training data. In theory, MLE could achieve optimal performance with infinite training data, but in practice this is impossible as the available resources are always limited. In particular, when the training data are scarce such as in low-resource language pairs or specific translation domains, NMT models display a modest performance, and other traditional approaches (e.g., PBSMT)[Koehn and Knowles, 2017] often obtain better accuracies. As such, generalization of NMT systems still calls for significant improvement.

We have proposed two novel regularization terms based on regressing word embeddings (ReWE) and regressing sentence embeddings (ReSE). ReWE is a module added to the decoder of a sequence-to-sequence model so that, during training, the model is trained to jointly predict the next word in the translation (categorical value) and its pre-trained word embedding (continuous value). This approach can leverage the contextual information embedded in pre-trained word vectors to achieve more accurate translations at test time. ReSE is an additional regularization method to further improve the accuracy of the translations. ReSE uses a self-attention mechanism to infer a fixed-dimensional sentence vector for the target sentence. During training, the model is trained to regress this inferred vector towards the pre-trained sentence embedding of the ground-truth sentence. Our main contributions in this chapter are:

- The proposal of a new regularization technique for NMT based on sentence embeddings (ReSE).

- Extensive experimentation over four language pairs of different dataset sizes (from small to large) with both word and sentence regularization. We show that using both ReWE and ReSE can outperform strong state-of-the-art baselines based on long short-term memory networks (LSTMs) and transformers.

- Insights on how ReWE and ReSE help to improve NMT models. Our analysis shows that these regularizers improve the organization of the decoder's output vector space, likely facilitating correct word classification.

- Further experimentation of the regularizer on *unsupervised* machine translation, showing that it can improve the quality of the translations even in the absence of parallel training data.

5.2 Related Work

The related work is organized over the three main research subareas that have motivated this work: *regularization techniques*, *word and sentence embeddings* and *unsupervised NMT*.

5.2.1 Regularization Techniques

In recent years, the research community has dedicated much attention to the problem of overfitting in deep neural models. Several regularization approaches have been proposed in turn such as dropout [Srivastava *et al.*, 2014; Gal and Ghahramani, 2016], data augmentation [Fadaee *et al.*, 2017] and multi-task learning [Gu *et al.*, 2018; Clark *et al.*, 2018]. Their common aim is to encourage the model to learn parameters that allow for better generalization.

In NMT, too, mitigating overfitting has been the focus of much research. As mentioned above, the two, main acknowledged problems are the single ground-truth reference and the exposure bias. For the former, Fadee et al. [2017] have proposed augmenting the training data with synthetically-generated sentence pairs containing rare words. The intuition is that the model will be able to see the vocabulary's words in more varied contexts during training. Kudo [Kudo, 2018] has proposed using variable word segmentations to improve the model's robustness, achieving notable improvements in low-resource languages and out-of-domain settings. Another line of work has focused on "smoothing" the output probability distribution over the target vocabulary [Elbayad *et al.*, 2018; Chousa *et al.*, 2018]. These approaches use token-level and sentence-level reward functions that push the model to distribute the output probability mass over words other than the ground-truth reference. Similarly, Ma et al. [Ma *et al.*, 2018] have added a bag-of-words term to the training objective, assuming that the set of correct translations share similar bag-of-word vectors.

There has also been extensive work on addressing the exposure bias problem. An approach that has proved effective is the incorporation of predictions in the training, via either imitation learning [Daumé *et al.*, 2009; Ross *et al.*, 2011; Leblond *et al.*, 2018] or reinforcement learning [Ranzato *et al.*, 2015; Bahdanau *et al.*, 2017]. Another approach, that is computationally more efficient, leverages scheduled sampling to obtain a stochastic mixture of words from the reference and the predictions [Bengio *et al.*, 2015]. In turn, Xu et al. [Xu *et al.*, 2019] have proposed a soft alignment algorithm to alleviate the missmatches between the reference translations and the predictions obtained with scheduled sampling; and Zhang et al.[Zhang *et al.*, 2018] have introduced two regularization terms based on the Kullback-Leibler (KL) divergence to improve the agreement of sentences predicted from left-to-right and right-to-left.

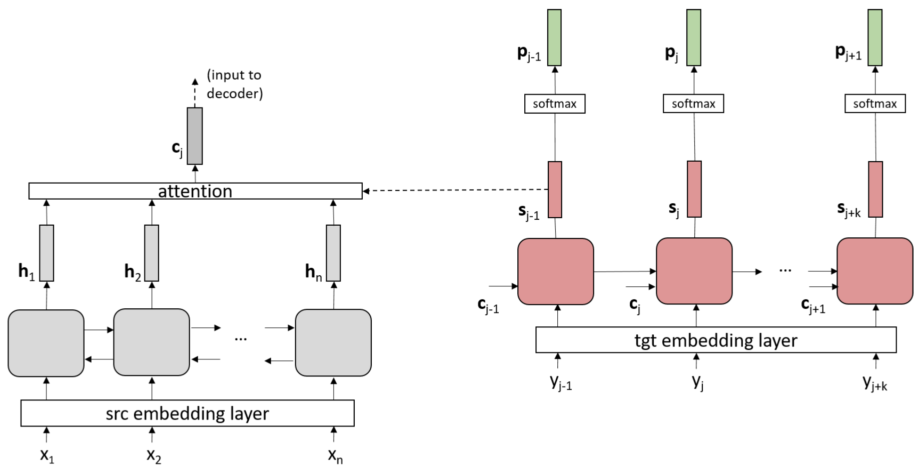


Figure 5.1: Baseline NMT model. (Left) The encoder receives the input sentence and generates a context vector $\mathbf{c}_j$ for each decoding step using an attention mechanism. (Right) The decoder generates one-by-one the output vectors $\mathbf{p}_j$, which represent the probability distribution over the target vocabulary. During training $\mathbf{y}_j$ is a token from the ground truth sentence, but during inference the model uses its own predictions.

5.2.2 Word and Sentence Embeddings

Word vectors or word embeddings [Mikolov *et al.*, 2013b; Pennington *et al.*, 2014; Bojanowski *et al.*, 2017] are ubiquitous in NLP since they provide effective input features for deep learning models. Recently, contextual word vectors such as ELMo [Peters *et al.*, 2018], BERT [Devlin *et al.*, 2019] and the OpenAI transformer [Radford *et al.*, 2018] have led to remarkable performance improvements in several language understanding tasks. Additionally, researchers have focused on developing embeddings for entire sentences and documents as they may facilitate several textual classification tasks [Kiros *et al.*, 2015; Conneau *et al.*, 2017; Cer *et al.*, 2018; Artetxe and Schwenk, 2019b].

In NMT models, word embeddings play an important role as input of both the encoder and the decoder. A recent paper has shown that contextual word embeddings provide effective input features for both stages [Edunov *et al.*, 2019]. However, very little research has been devoted to using word embeddings as targets. Kumar and Tsvetkov [Kumar and Tsvetkov, 2018] have removed the typical output softmax layer, forcing the decoder to generate continuous outputs. At inference time, they use a nearest-neighbour search in the word embedding space to select the word to predict. Their model allows for significantly faster training while performing on par with state-of-the-art models. Our approach differs from [Kumar and Tsvetkov, 2018] in that our decoder generates continuous outputs *in parallel* with the standard softmax layer, and only during training to provide regularization. At inference time, the continuous output is ignored and prediction operates as in a standard NMT model. To the best of our knowledge, our model is the first to use embeddings as targets for regularization, and at both word and sentence level.

5.2.3 Unsupervised NMT

The amount of available parallel, human-annotated corpora for training NMT systems is at times very scarce. This is the case of many low-resource languages and specialized translation domains (e.g., health care). Consequently, there has been a growing interest in developing unsupervised NMT models [Lample *et al.*, 2018; Artetxe *et al.*, 2018; Yang *et al.*, 2018] which do not require annotated data for training. Such models learn to translate by only using monolingual corpora, and even though their accuracy is still well below that of their supervised counterparts, they have started to reach interesting levels. The architecture of unsupervised NMT systems differs from that of supervised systems in that it combines translation in both directions (source-to-target and target-to-source). Typically, a single encoder is used to encode sentences from both languages, and a separate decoder generates the translations in each language. The training of such systems follows three stages: 1) building a bilingual dictionary and word embedding space, 2) training two monolingual language models as denoising autoencoders [Vincent *et al.*, 2008], and 3) converting the unsupervised problem into a weakly-supervised one by use of back-translations [Sennrich *et al.*, 2016]. For more details on unsupervised NMT systems, we refer the reader to the original papers [Lample *et al.*, 2018; Artetxe *et al.*, 2018; Yang *et al.*, 2018].

In this chapter, we explore using the proposed regularization approach also for unsupervised NMT. Unsupervised NMT models still require very large amounts of monolingual data for training, and often such amounts are not available. Therefore, these models, too, are expected to benefit from improved regularization.

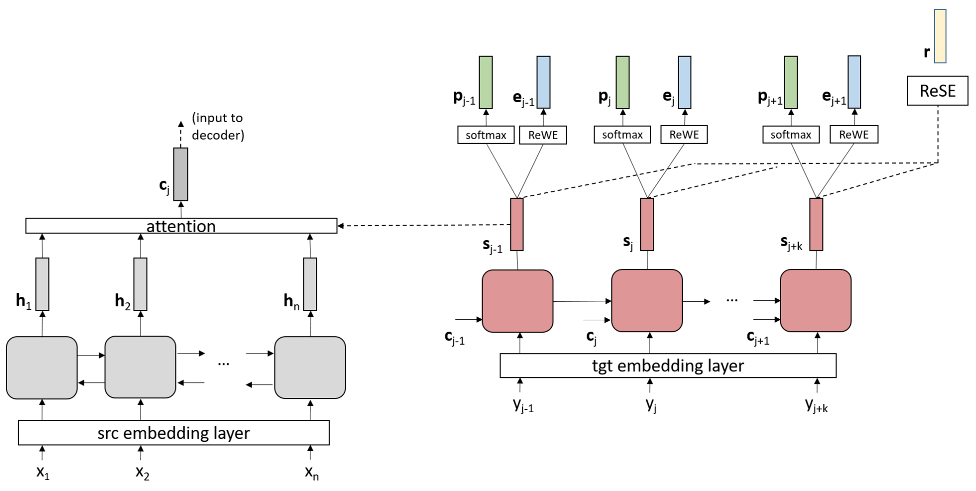


Figure 5.2: Full model: Baseline + ReWE + ReSE. (Left) The encoder with the attention mechanism generates vectos $\mathbf{c}_j$ in the same way as the baseline system. (Right) The decoder generates one-by-one the output vectors $\mathbf{p}_j$, which represent the probability distribution over the target vocabulary, and $\mathbf{e}_j$, which is a continuous word vector. Additionally, the model can also generate another continuous vector, $\mathbf{r}$, which represents the sentence embedding.

5.3 The Baseline NMT model

In this section, we describe the NMT model that has been used as the basis for the proposed regularizer. It is a neural encoder-decoder architecture with attention [Bahdanau *et al.*, 2015] that can be regarded as a strong baseline as it incorporates both LSTMs and transformers as modules. Let us assume that $\mathbf{x} : \{x_1 \dots x_n\}$ is the source sentence with n tokens and $\mathbf{y} : \{y_1 \dots y_m\}$ is the target translated sentence with m tokens. First, the words in the source sentence are encoded into their word embeddings by an embedding layer:

$$\mathbf{x}_i^e = SrcEmbLayer(x_i) \quad i = 1 \dots n \tag{5.1}$$

and then the source sentence is encoded by a sequential module into its hidden vectors, $\mathbf{h}_1 \dots \mathbf{h}_n$:

$$\mathbf{h}_i = enc(\mathbf{h}_{i-1}, \mathbf{x}_i^e) \quad i = 1 \dots n \tag{5.2}$$

Next, for each decoding step $j = 1 \dots m$, an attention network provides a context vector $\mathbf{c}_j$ as a weighted average of all the encoded vectors, $\mathbf{h}_1 \dots \mathbf{h}_n$, conditional on the decoder output at the previous step, $\mathbf{s}_{j-1}$ (Eq. 5.3). For this network, we have used the attention mechanism of Badhdanau et al.[Bahdanau *et al.*, 2015].

$$\mathbf{c}_j = attn(\mathbf{h}_1 \dots \mathbf{h}_n, \mathbf{s}_{j-1}) \quad j = 1 \dots m \tag{5.3}$$

Given the context vector, $\mathbf{c}_j$, the decoder output at the previous step, $\mathbf{s}_{j-1}$, and the word embedding of the previous word in the target sentence, $\mathbf{y}^e_j$ (Eq. 5.4) (Eq. 5.5), the decoder generates vector $\mathbf{s}_j$, where $\mathbf{s}_0$ is a fixed vector that indicates the start of a sentence. This vector is later transformed into a larger vector of the same size as the target vocabulary via learned parameters $\mathbf{W}$, $\mathbf{b}$ and a softmax layer (Eq. 5.6). The resulting vector, $\mathbf{p}_j$, is the inferred probability distribution over the target vocabulary at decoding step j. Fig. 5.1 depicts the full architecture of the baseline model.

$$\mathbf{y}^e_j = TgtEmbLayer(y_j) \quad j = 1 \dots m \tag{5.4}$$

$$\mathbf{s}_j = dec(\mathbf{c}_j, \mathbf{s}_{j-1}, \mathbf{y}^e_{j-1}) \quad j = 1 \dots m \tag{5.5}$$

$$\mathbf{p}_j = softmax(\mathbf{W}\mathbf{s}_j + \mathbf{b}) \tag{5.6}$$

The model is trained by minimizing the negative log-likelihood (NLL) which can be expressed as:

$$\mathcal{L}_{NLL} = -\sum_{j=1}^{m} \log(\mathbf{p}_j(y_j)) \tag{5.7}$$

where the probability of ground-truth word y_j has been noted as $\mathbf{p}_j(y_j)$. Minimizing the NLL is equivalent to MLE and results in assigning maximum probability to the words in the reference translation, $y_j, j = 1 \dots m$. The training objective is minimized with standard backpropagation over the training data, and at inference time the model uses beam search for decoding.

5.4 Regressing word and sentence embeddings

As mentioned in the introduction, MLE suffers from some limitations when training a neural machine translation system. To alleviate these shortcomings, we have proposed two new regularization terms for the objective function based on regressing word and sentence embeddings.

5.4.1 ReWE

Pre-trained word embeddings are trained on large monolingual corpora by measuring the co-occurences of words in text windows ("contexts"). Words that occur in similar contexts are assumed to have similar meaning, and hence, similar vectors in the embedding space. Our goal with ReWE is to incorporate the information embedded in the word vector in the loss function to encourage model regularization.

In order to generate continuous vector representations as outputs, we have added a ReWE block to the NMT baseline (Fig. 5.2). At each decoding step, the ReWE block receives the hidden vector from the decoder, $\mathbf{s}_j$, as input and outputs another vector, $\mathbf{e}_j$, of the same size of the pre-trained word embeddings:

$$\begin{aligned}\mathbf{e}_j &= ReWE(\mathbf{s}_j) \\ &= \mathbf{W}_2(ReLU(\mathbf{W}_1\mathbf{s}_j + \mathbf{b}_1)) + \mathbf{b}_2\end{aligned} \tag{5.8}$$

where $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$ and $\mathbf{b}_2$ are the learnable parameters of a two-layer feed-forward network with a Rectified Linear Unit (ReLU) as activation function between the layers. Vector $\mathbf{e}_j$ aims to reproduce the word embedding of the target word, and thus the distributional properties (or co-occurrences) of its contexts.

During training, the model is guided to regress the predicted vector, $\mathbf{e}_j$, towards the word embedding of the ground-truth word, $\mathbf{y}^e_j$. This is achieved by using a loss function that computes the distance between $\mathbf{e}_j$ and $\mathbf{y}^e_j$. In the experiment, we have explored two alternative distance metrics for the $ReWE_{loss}$: the minimum square error (MSE)[1] and the cosine embedding loss (CEL) [2]. This loss and the original NLL loss are combined together with a tunable hyper-parameter, λ (Eq. 5.9). Therefore, the model is trained to jointly predict both a categorical and a continuous representation of the words. Even though the system is performing a single task, this setting could also be interpreted as a form of multi-task learning with different representations of the same targets.

$$\mathcal{L}_w = \mathcal{L}_{NLL} + \lambda \mathcal{L}_{ReWE} \tag{5.9}$$

The word vectors of both the source ($\mathbf{x}^e$) and target ($\mathbf{y}^e$) vocabularies are initialized with pre-trained embeddings, but updated during training. At inference time, we ignore the outputs of the ReWE block and we perform translation using only the categorical prediction.

5.4.2 ReSE

Sentence vectors, too, have been extensively used as input representations in many NLP tasks such as text classification, paraphrase detection, natural language inference and question answering. The intuition behind them is very similar to that of word embeddings: sentences with similar meanings are expected to be close to each other in vector space. Many off-the-shelf sentence embedders are currently available and they can be easily integrated in deep learning models. Based on similar assumptions to the case of

[1]https://pytorch.org/docs/stable/nn.html#torch.nn.MSELoss

[2]https://pytorch.org/docs/stable/nn.html#torch.nn.CosineEmbeddingLoss

word embeddings, we have hypothesized that an NMT model could also benefit from a regularization term based on regressing sentence embeddings (the ReSE block in Fig. 5.2).

The main difference of ReSE compared to ReWE is that there has to be a single regressed vector per sentence rather than one per word. Thus, ReSE first uses a self-attention mechanism to learn a weighted average of the decoder's hidden vectors, $\mathbf{s}_1 \dots \mathbf{s}_m$:

$$SelfAttn(\mathbf{s}_1, \dots, \mathbf{s}_m) = \sum_{j=0}^{m} \alpha_j \mathbf{s}_j \tag{5.10}$$

$$\alpha_j = \frac{e^{l_j}}{\sum_{k=0}^{m} e^{l_k}} \tag{5.11}$$

$$l_j = \mathbf{U}_2 \tanh(\mathbf{U}_1 \mathbf{s}_j) \tag{5.12}$$

where the α_j attention weights are obtained from Eqs. 5.11 and 5.12, and $\mathbf{U}_1$ and $\mathbf{U}_2$ are learnable parameters. Then, a two-layered neural network similar to ReWE's predicts the sentence vector, $\mathbf{r}$ (Eq. 5.13). Parameters $\mathbf{W}_3$, $\mathbf{W}_4$, $\mathbf{b}_3$ and $\mathbf{b}_4$ are also learned during training.

$$\begin{aligned}\mathbf{r} &= ReSE([\mathbf{s}_1, \dots, \mathbf{s}_m]) \\ &= \mathbf{W}_3(ReLU(\mathbf{W}_4 SelfAttn([\mathbf{s}_1, \dots, \mathbf{s}_m]) + \mathbf{b}_3)) + \mathbf{b}_4\end{aligned} \tag{5.13}$$

Similarly to ReWE, a loss function computes the distance between the predicted sentence vector, $\mathbf{r}$, and the sentence vector inferred with the off-the-shelf sentence embedder, $\mathbf{y}^r$. This loss is added to the previous objective as an extra term with an additional, tunable hyper-parameter, β:

$$\mathcal{L}_{ws} = \mathcal{L}_{NLL} + \lambda \mathcal{L}_{ReWE} + \beta \mathcal{L}_{ReSE} \tag{5.14}$$

Since the number of sentences is significantly lower than that of the words, β typically needs to be higher than λ. Nevertheless, we tune it blindly using the validation set. The reference sentence embedding, $\mathbf{y}^r$, can be inferred with any off-the-shelf pre-trained embedder. At inference time, the model solely relies on the categorical prediction and ignores the predicted word and sentence vectors.

5.5 Experiments

We have carried out an ample range of experiments to probe the performance of the proposed regularization approaches. This section describes the datasets, the models and the hyper-parameters used, and presents and discusses all results.

5.5.1 Datasets

Four different language pairs have been selected for the experiments. The datasets' size varies from tens of thousands to millions of sentences to test the regularizers' ability to improve translation over a range of low-resource and high-resource language pairs.

De-En: The German-English dataset (de-en) has been taken from the WMT18 news translation shared task[3]. The training set contains over 5M sentence pairs collected from the *Europarl*, *CommonCrawl* and *Newscommentary* parallel corpora. As validation and test sets, we have used the *newstest2017* and the *newstest2018* datasets, respectively. We consider this dataset as a high-resource case.

En-Fr: The English-French dataset (en-fr) has been sourced from the IWSLT 2016 translation shared task[4]. This corpus contains translations of TED talks of very diverse topics. The training data provided by the organizers consist of $219,777$

[3]WMT18: http://www.statmt.org/wmt18/translation-task.html
[4]IWSLT16: https://workshop2016.iwslt.org/

Language pair	Dataset	Train	Dev	Test
De-En	WMT18 News	5M	6K	3K
En-Fr	IWSLT16 TED talks	220K	2K	2K
Cs-En	IWSLT16 TED talks	110K	3K	2K
Eu-En	WMT16 IT-domain	90K	1K	1K

Table 5.1: Approximate number of sentences in the each train, dev and test datasets.

translations which allow us to categorize this dataset as low/medium-resource. Following Denkowski and Neubig [Denkowski and Neubig, 2017], the validation set has been formed by merging the 2013 and 2014 test sets from the same shared task, and the test set has been formed with the 2015 and 2016 test sets.

Cs-En: The Czech-English dataset (cs-en) is also from the IWSLT 2016 TED talks translation task. However, this dataset is approximately half the size of en-fr as its training set consists of $114,243$ sentence pairs. Again following Denkowski and Neubig [Denkowski and Neubig, 2017]), the validation set has been formed by merging the 2012 and 2013 test sets, and the test set by merging the 2015 and 2016 test sets. We regard this dataset as low-resource.

Eu-En: The Basque-English dataset (eu-en) has been collected from the WMT16 IT-domain translation shared task[5]. This is the smallest dataset, with only $89,413$ sentence pairs in the training set. However, only $2,000$ sentences in the training set have been translated by human annotators. The remaining sentence pairs are translations of IT-domain short phrases and Wikipedia titles. Therefore, we consider this dataset as extremely low-resource. It must be said that translations in the IT domain are somehow easier than in the news domain, as this domain is very specific and the wording of the sentences are less varied. For this dataset, we have used the validation and test sets ($1,000$ sentences each) provided in the shared task.

All the datasets have been pre-processed with *moses-tokenizer*[6]. Additionally, words have been split into subword units using byte pair encoding (BPE) [Sennrich *et al.*, 2015]. For the BPE merge operations parameter, we have used $32,000$ (the default value) for

[5]WMT16 IT: http://www.statmt.org/wmt16/it-translation-task.html
[6]URL moses

all the datasets, except for eu-en where we have set it to $8,000$ since this dataset is much smaller. Experiments have been performed at both word and subword level since morphologically-rich languages such as German, Czech and Basque can benefit greatly from operating the NMT model at subword level. When using words as atomic units, we have limited the vocabulary to the most frequent 50,000 tokens in the training data.

5.5.2 Model Training and Hyper-Parameter Selection

To implement ReWE and ReSE, we have modified the popular OpenNMT open-source toolkit [Klein *et al.*, 2017][7]. Two variants of the standard OpenNMT model have been used as baselines: the LSTM and the transformer, described hereafter.

LSTM: A strong NMT baseline was prepared by following the indications given by Denkowski and Neubig [Denkowski and Neubig, 2017]. The model uses a bidirectional LSTM [Hochreiter and Schmidhuber, 1997] for the encoder and a unidirectional LSTM for the decoder, with two layers each. The size of the word embeddings was set to 300d and that of the sentence embeddings to 512d. The sizes of the hidden vectors of both LSTMs and of the attention network were set to 1024d. In turn, the LSTM's dropout rate was set to 0.2 and the training batch size was set to 40 sentences. As optimizer, we have used Adam [Kingma and Ba, 2015] with a learning rate of 0.001. During training, the learning rate was halved with simulated annealing upon convergence of the perplexity over the validation set, which was evaluated every $25,000$ training sentences. Training was stopped after halving the learning rate 5 times.

Transformer: The transformer network [Vaswani *et al.*, 2017] has somehow become the *de-facto* neural network for the encoder and decoder of NMT pipelines thanks to its strong empirical accuracy and highly-parallelizable training. For this reason, we have used it as another baseline for our model. For its hyper-parameters, we have used the default values set by the developers of OpenNMT[8]. Both the encoder and the decoder are formed by a 6-layer network. The sizes of the word

[7]Our code is publicly available on Github at: https://github.com/ijauregiCMCRC/ReWE_and_ReSE.git. We will also release it on CodeOcean.

[8]Transformer: http://opennmt.net/OpenNMT-py/FAQ.html

embeddings, the hidden vectors and the attention network have all been set to either 300d or 512d, depending on the best results over the validation set. The head count has been set correspondingly to either 6 or 8, and the dropout rate to 0.2 as for the LSTM. The model was also optimized using Adam, but with a much higher learning rate of 1 (OpenAI default). For this model, we have not used simulated annealing since some preliminary experiments showed that it did penalize performance. The batch size used was $4,096$ and $1,024$ words, again selected based on the accuracy over the validation set. Training was stopped upon convergence in perplexity over the validation set, which was evaluated at every epoch.

In addition, the word embeddings for both models were initialized with pre-trained fastText embeddings [Bojanowski *et al.*, 2017]. For the 300d word embeddings, we have used the word embeddings available on the official fastText website[9]. For the 512d embeddings and the subword units, we have trained our own pre-trained vectors using the fastText embedder with a large monolingual corpora from Wikipedia[10] and the training data. Both models have used the same sentence embeddings which have been computed with the Universal Sentence Encoder (USE)[11]. However, the USE is only available for English, so we have only been able to use ReSE with the datasets where English is the target language (i.e., de-en, cs-en and eu-en). When using BPE, the subwords of every sentence have been merged back into words before passing them to the USE. The BLEU score for the BPE models has also been computed after post-processing the subwords back into words. Finally, hyper-parameters λ and β have been tuned only once for all datasets by using the en-fr validation set. This was done in order to save the significant computational time that would have been required by further hyper-parameter exploration. However, in the de-en case the initial results were far from the state of the art and we therefore repeated the selection with its own validation set. For all experiments, we have used an Intel Xeon E5-2680 v4 with an NVidia GPU card Quadro P5000. On this machine, the training time of the transformer has been approximately an order of magnitude larger than that of the LSTM.

[9]Fasttext: https://fasttext.cc/docs/en/crawl-vectors.html
[10]Wikipedia: https://linguatools.org/tools/corpora/
[11]USE: https://tfhub.dev/google/universal-sentence-encoder/2

Models	Word/BPE	BLEU
LSTM	word	34.21
LSTM + ReWE($\lambda = 20$)	word	35.43†
Transformer	word	34.56
Transformer + ReWE($\lambda = 20$)	word	35.3†
LSTM	bpe	34.06
LSTM + ReWE($\lambda = 20$)	bpe	35.09†
Transformer	bpe	35.31
Transformer + ReWE($\lambda = 20$)	bpe	**36.3**†

Table 5.2: BLEU scores over the En-Fr test set. The reported results are the average of 5 independent runs. (†) means that the differences are statistically significant with respect to the baseline with a p-value < 0.05 over a two-tailed Welch's t-test.

Models	Word/BPE	BLEU
LSTM	word	20.48
+ ReWE($\lambda = 20$)	word	21.81†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	word	21.98†
Transformer	word	20.56
+ ReWE($\lambda = 20$)	word	21.16†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	word	20.05†
LSTM	bpe	22.56
+ ReWE($\lambda = 20$)	bpe	**23.72**†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	bpe	23.56†
Transformer	bpe	21.02
+ ReWE($\lambda = 20$)	bpe	22.19†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	bpe	20.53

Table 5.3: BLEU scores over the Cs-En test set. The reported results are the average of 5 independent runs. (†) means that the differences are statistically significant with respect to the baseline with a p-value < 0.05 over a two-tailed Welch's t-test.

5.5.3 Results

We have carried out a number of experiments with both baselines. The scores reported are an average of the BLEU scores (in percentage points, or pp) [Papineni *et al.*, 2002] over the test sets of 5 independently trained models. Table 5.2 shows the results over the en-fr dataset. In this case, the models with ReWE have outperformed the LSTM and transformer baselines consistently. The LSTM did not benefit from using BPE, but the transformer+ReWE with BPE reached 36.30 BLEU pp (a $+0.99$ pp improvement over the best model without ReWE). For this dataset we did not use ReSE because French was the target language.

Models	Word/BPE	BLEU
LSTM	word	10.87
+ ReWE($\lambda = 20$)	word	13.83†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	word	16.02†
Transformer	word	12.15
+ ReWE($\lambda = 20$)	word	13.53†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	word	6.92
LSTM	bpe	17.14
+ ReWE($\lambda = 20$)	bpe	19.54†
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	bpe	**20.29**†
Transformer	bpe	12.70
+ ReWE($\lambda = 20$)	bpe	13.21
+ ReWE($\lambda = 20$) + ReSE($\beta = 100$)	bpe	9.63

Table 5.4: BLEU scores over the Eu-En test set. The reported results are the average of 5 independent runs. (†) means that the differences are statistically significant with respect to the baseline with a p-value < 0.05 over a two-tailed Welch's t-test.

Models	Word/BPE	BLEU
LSTM	word	29.75
+ ReWE($\lambda = 2$)	word	30.17†
+ ReWE($\lambda = 2$) + ReSE($\beta = 2$)	word	30.23†
LSTM	bpe	**34.03**
+ ReWE($\lambda = 2$)	bpe	33.66
+ ReWE($\lambda = 2$) + ReSE($\beta = 2$)	bpe	33.91

Table 5.5: BLEU scores over the De-En test set. The reported results are the average of 5 independent runs. (†) means that the differences are statistically significant with respect to the baseline with a p-value < 0.05 over a two-tailed Welch's t-test.

Table 5.3 reports the results over the cs-en dataset. Also in this case, all the models with ReWE have improved over the corresponding baselines. The LSTM+ReWE has achieved the best results (23.72 BLEU pp; an improvement of $+1.16$ pp over the best model without ReWE). This language pair has also benefited more from the BPE pre-processing, likely because Czech is a morphologically-rich language. For this dataset, it was possible to use ReSE in combination with ReWE, with an improvement for the LSTM at word level ($+0.14$ BLEU pp), but not for the remaining cases. We had also initially tried to use ReSE without ReWE (i.e., $\lambda = 0$), but the results were not encouraging and we did not continue with this line of experiments.

For the eu-en dataset (Table 5.4), the results show that, again, ReWE outperforms the baselines by a large margin. Moreover, ReWE+ReSE has been able to improve the results even further ($+3.15$ BLEU pp when using BPE and $+5.15$ BLEU pp at word level over the corresponding baselines). Basque is, too, a morphologically-rich language and using BPE has proved very beneficial ($+4.27$ BLEU pp over the best word-level model). As noted before, the eu-en dataset is very low-resource (less than $100,000$ sentence pairs) and it is more likely that the baseline models generalize poorly. Consequently, regularizers such as ReWE and ReSE are more helpful, with larger margins of improvement with respect to the baselines. On a separate note, the transformer has unexpectedly performed well below the LSTM on this dataset, and especially so with BPE. We speculate that it may be more sensitive than the LSTM to the dataset's much smaller size, or in need of more refined hyper-parameter tuning.

Finally, Table 5.5 shows the results over the de-en dataset that we categorize as high-resource (5M+ sentence pairs). For this dataset, we have only been able to perform experiments with the LSTM due to the exceedingly long training times of the transformer. At word level, both ReWE and ReWE+ReSE have been able to outperform the baseline, although the margins of improvement have been smaller than for the other language pairs ($+0.42$ and $+0.48$ BLEU pp, respectively). However, when using BPE both ReWE and ReWE+ReSE have performed slightly below the baseline (-0.37 and -0.12 points BLEU pp, respectively). This shows that when the training data are abundant, ReWE or ReSE may not be beneficial. To probe this further, we have repeated these experiments by training the models over subsets of the training set of increasing size (200K, 500K, 1M, and 2M sentence pairs). Fig. 5.3 shows the BLEU scores achieved by the baseline and the regularized models for the different training data sizes. The plot clearly shows that the performance margin increases as the training data size decreases, as expected from a regularized model.

As a preliminary experiment, we have carried out a sensitivity analysis to determine the optimal value of the trade-off coefficient, λ, using the en-fr validation set. The results are shown in Figure 5.4, where each point is the average of three runs trained with different seeds. The figure shows that the MSE loss has outperformed slightly the baseline for small values of $lambda$ (< 1), but the BLEU score has dropped drastically for larger values. Conversely, the CEL loss has increased steadily with λ, reaching 38.23 BLEU points for λ = 20, with a marked improvement of 1.53 points over the baseline. This re-

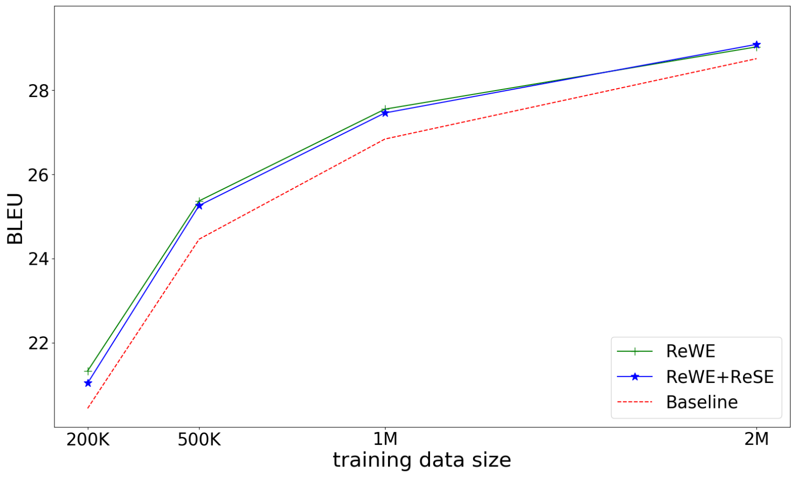


Figure 5.3: BLEU scores over the de-en test set for models trained with training sets of different size.

sult has been encouraging and therefore for the rest of the experiments we have used CEL as the ReWE loss and kept the value of λ to 20. A similar analysis has been done to tune β, using only the CEL distance. The results of this analysis over the Cs-En dataset are shown in Figure 5.5.

Table 5.7 shows two examples of the translations made by the different LSTM models for eu-en and cs-en. A qualitative analysis of these examples shows that both ReWE and ReWE+ReSE have improved the quality of these translations. In the eu-en example, ReWE has correctly translated “File tab”; and ReSE has correctly added “click Create”. In the cs-en example, the model with ReWE has picked the correct subject “they”, and only the model with ReWE and ReSE has correctly translated “students” and captured the opening phrase “What was. . . about this. . . ”.

5.5.4 Understanding ReWE and ReSE

The quantitative experiments have proven that ReWE and ReSE can act as effective regularizers for low- and medium-resource NMT. Yet, it would be very interesting to understand how do they influence the training to achieve improved models. For that purpose, we have conducted an exploration of the values of the hidden vectors on the decoder end ($\mathbf{s}_j$, Eq. 5.5). These values are the “feature space” used by the final classification block (a linear transformation and a softmax) to generate the class probabilities and can provide insights on the model. For this reason, we have considered the cs-en test set and stored all

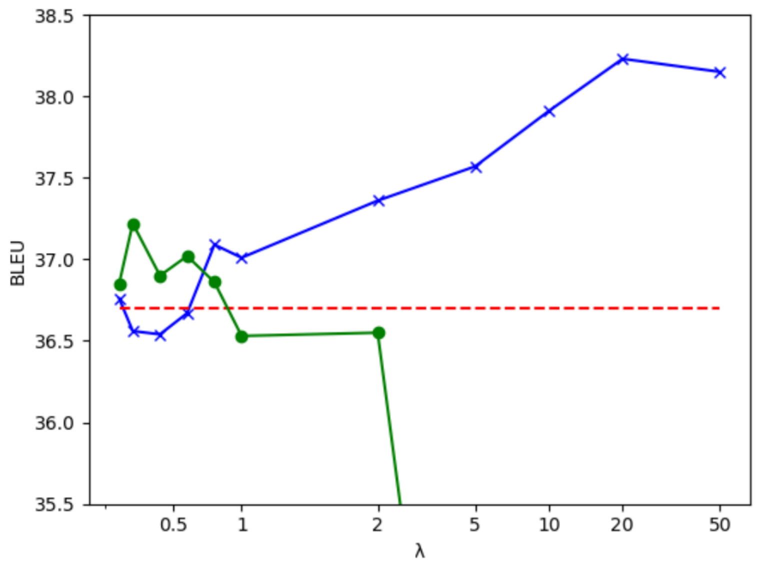


Figure 5.4: BLEU scores of three models over the enfr validation set for different λ values: baseline (red), baseline + ReWE (MSE) (green), baseline + ReWE (CEL) (blue). Each point in the graph is an average of 3 independently trained models.

the $\mathbf{s}_j$ vectors with their respective word predictions. Then, we have used t-SNE [Maaten and Hinton, 2008] to reduce the dimensionality of the $\mathbf{s}_j$ vectors to a visualizable 2d. Finally, we have chosen a particular word (*architecture*) as the center of the visualization, and plotted all the vectors within a chosen neighborhood of this center word (Fig. 5.6). To avoid cluttering the figure, we have not superimposed the predicted words to the vectors, but only used a different color for each distinct word. The center word in the two subfigures (a: baseline; b: baseline+ReWE) is the same (*architecture*) and from the same source sentence, so the visualized regions are comparable. The visualizations also display all other predicted instances of word *architecture* in the neighborhood.

These visualizations show two interesting behaviors: 1) from eye judgment, the points predicted by the ReWE model seem more uniformly spread out; 2) instances of the same words have $\mathbf{s}_j$ vectors that are close to each other. For instance, several instances of word *architecture* are close to each other in Fig. 5.6b while a single instance appears in Fig. 5.6b. The overall observation is that the ReWE regularizer leads to a vector space that is easier to discriminate, i.e. find class boundaries for, facilitating the final word prediction. In order to confirm this observation, we have computed various clustering indexes over the clusters formed by the vectors with identical predicted word. As indexes, we have used the silhouette and the Davies-Bouldin indexes that are two well-known unsupervised

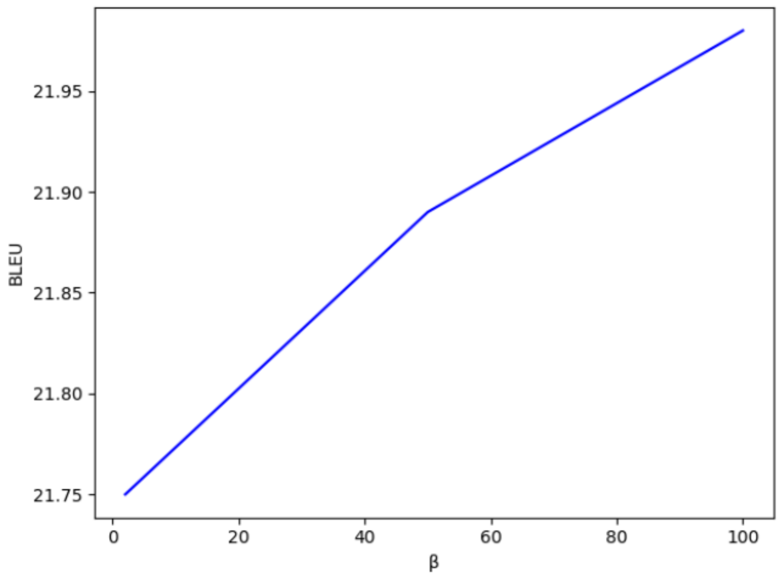


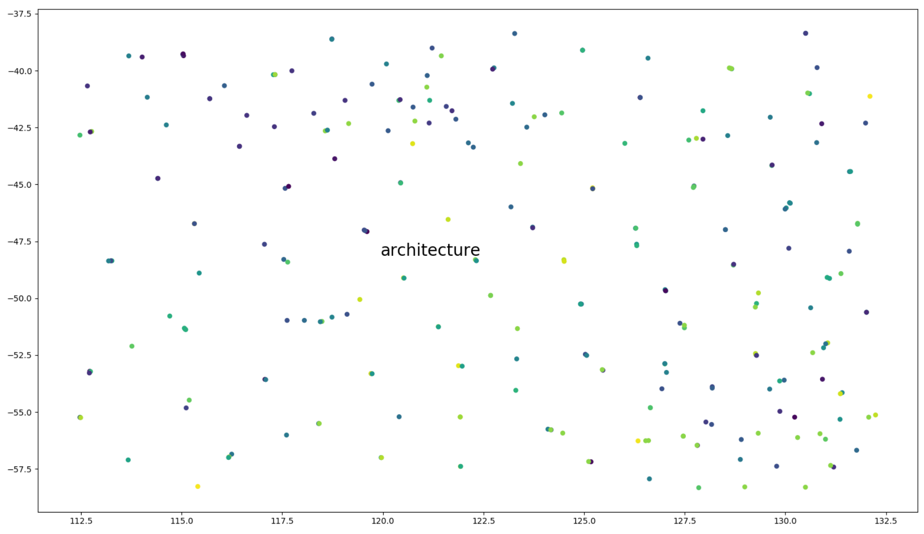
Figure 5.5: BLEU scores over the Cs-En dev set of a baseline + ReWE + ReSE model, with λ fixed to 20 and different β values. Each point in the graph is an average of 3 independently trained models.

Model	Sillhouette	Davies-Bouldin
LSTM	-0.19	1.87
+ ReWE($\lambda = 2$)	-0.17	**1.80**
+ ReWE($\lambda = 2$) + ReSE($\beta = 2$)	**-0.16**	**1.80**

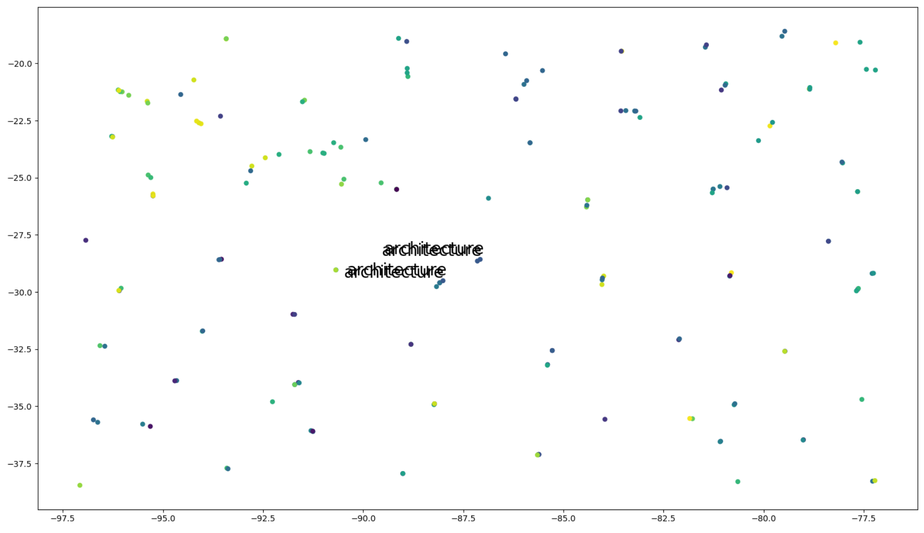
Table 5.6: Clustering indexes of the LSTM models over the cs-en test set. The reported results are the average of 5 independent runs.

metrics for clustering. The silhouette index ranges from -1 to +1, where values closer to 1 mean that the clusters are compact and well separated, and hence more desirable. The Davies-Bouldin index is an unbounded non-negative value, with values closer to 0 meaning better clustering. Table 5.6 shows the values of these clustering indexes over the entire cs-en test set for the LSTM models. As the table shows, the models with ReWE and ReWE+ReSE have reported the best values. This confirms that applying ReWE and ReSE has a positive impact on the decoder's hidden space, ultimately justifying the increase in word classification accuracy.

For further exploration, we have created another visualization of the $\mathbf{s}$ vectors and their predictions over a smaller neighborhood (Fig. 5.7). The same word (*architecture*) has been used as the center word of the plot. Then, we have "vibrated" each of the $\mathbf{s}_j$ vector by small increments (between 0.05 and 8 units) in each of their dimensions,



(a) Baseline



(b) Baseline + ReWE

Figure 5.6: Visualization of the $\mathbf{s}_j$ vectors from the decoder for a subset of the cs-en test set. Please refer to Section 5.5.4 for explanations. This figure should be viewed in color.

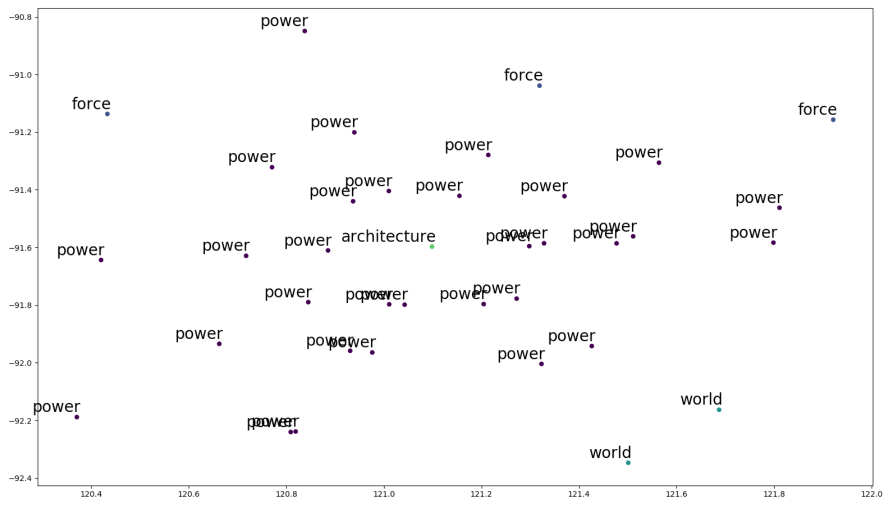
creating several new synthetic instances of **s** vectors which are very close to the original ones. These synthetic vectors have then been decoded with the trained NMT model to obtain their predicted words. Finally, we have used t-SNE to reduce the dimensionality to 2d, and visualized all the vectors and their predictions in a small neighborhood (± 10 units) around the center word. Fig. 5.7 shows that, with the ReWE model, all the **s** vectors surrounding the center word predict the same word (*architecture*). Conversely, with the baseline, the surrounding points predict different words (*power*, *force*, *world*). This is additional evidence that the **s** space is evened out by the use of the proposed regularizer.

5.5.5 Unsupervised NMT

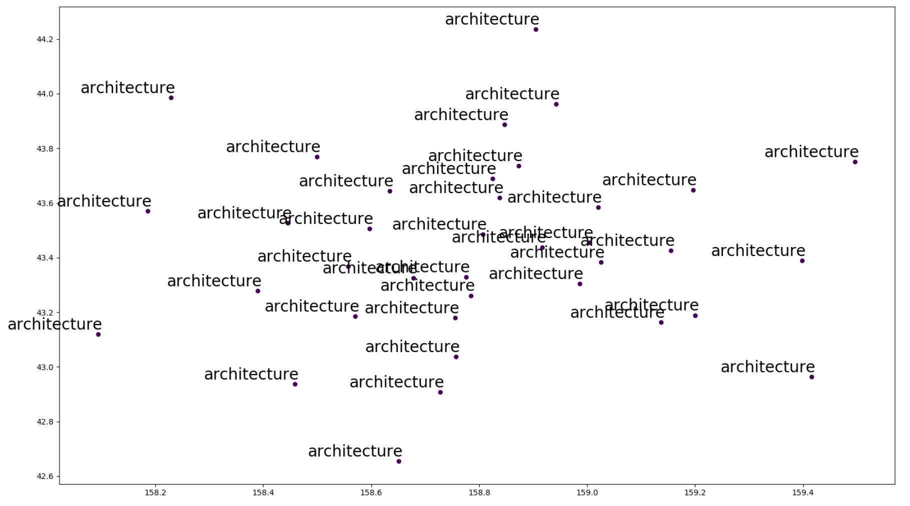
Finally, we have also experimented with the use of ReWE and ReWE+ReSE for an unsupervised NMT task. For this experiment, we have used the open-source model provided by Lample et al. [Lample *et al.*, 2018][12] which is currently the state of the art for unsupervised NMT, and also adopted its default hyper-parameters and pre-processing steps which include 4-layer transformers for the encoder and both decoders, and BPE subword learning. The experiments have been performed using the WMT14 English-French test set for testing in both language directions (en-fr and fr-en), and the monolingual data from that year's shared task for training.

As described in Section 5.2.3, an unsupervised NMT model contains two decoders to be able to translate into both languages. The model is trained by iterating over two alternate steps: 1) training using the decoders as monolingual, de-noising language models (e.g., en-en, fr-fr), and 2) training using back-translations (e.g., en-fr-en, fr-en-fr). Each step requires an objective function, which is usually an NLL loss. Moreover, each step is performed in both directions (en$\rightarrow$fr and fr$\rightarrow$en), which means that an unsupervised NMT model uses a total of four different objective functions. Potentially, the regularizers could be applied to each of them. However, the pre-trained USE sentence embeddings are only available in English, not in French, and for this reason we have limited our experiments to ReWE alone. In addition, the initial results have showed that ReWE is actually detrimental in the de-noising language model step, so we have limited its use to both language directions in the back-translation step, with the hyper-parameter, λ, tuned over the

[12]UnsupervisedMT: https://github.com/facebookresearch/UnsupervisedMT

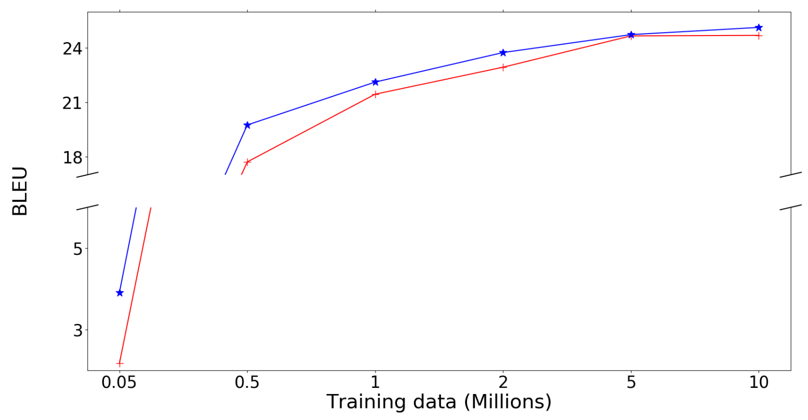


(a) Baseline

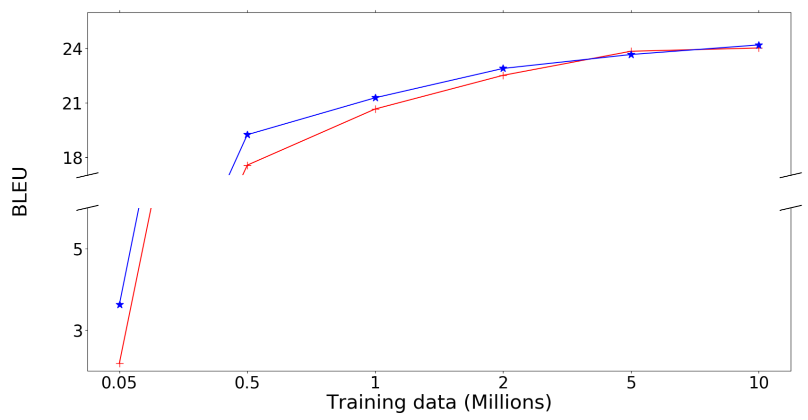


(b) Baseline + ReWE

Figure 5.7: Visualization of the $\mathbf{s}_j$ vectors in a smaller neighborhood of the center word.



(a) en-fr



(b) fr-en

Figure 5.8: BLEU scores over the test set. The reported results are the average of 5 independent runs.. The red line represents the baseline model and the blue line is the baseline + ReWE.

validation set ($\lambda = 0.2$).

To probe the effectiveness of the regularized model, Fig. 5.8 shows the results over the test set from the different models trained with increasing amounts of monolingual data (50K, 500K, 1M, 2M, 5M and 10M sentences in each language). The model trained using ReWE has been able to consistently outperform the baseline in both language directions. The trend we had observed in the supervised case has applied to these experiments, too: the performance margin has been larger for smaller training data sizes. For example, in the en-fr direction the margin has been $+1.74$ BLEU points with 50K training sentences, but it has reduced to $+0.44$ BLEU points when training with 10M sentences. Again, this behavior is in line with the regularizing nature of the proposed regressive objectives.

5.6 Conclusion

In this chapter, we have proposed regressing continuous representations of words and sentences (ReWE and ReSE, respectively) as novel regularization techniques for improving the generalization of NMT models. Extensive experiments over four different language pairs of different training data size (from 89K to 5M sentence pairs) have shown that both ReWE and ReWE+ReSE have improved the performance of NMT models, particularly in low- and medium-resource cases, for increases in BLEU score up to 5.15 percentage points. In addition, we have presented a detailed analysis showing how the proposed regularization modifies the decoder's output space, enhancing the clustering of the vectors associated with unique words. Finally, we have showed that the regularized models have also outperformed the baselines in experiments on unsupervised NMT. As future work, we plan to explore how the categorical and continuous predictions from our model could be jointly utilized to further improve the quality of the translations.

Example 1:	
Src:	Sakatu Fitxategia fitxa Oihal atzeko ikuspegia atzitzeko ; sakatu Berria . Hautatu txantiloia eta sakatu Sortu hautatutako txantiloia erabiltzeko .
Ref:	Click the File tab to access Backstage view , select New . Select a template and click Create to use the selected template .
Baseline:	Click the default tab of the tab that you want to open the tab tab . Select the template and select the selected template .
Baseline + ReWE:	Press the File tab to access the view view ; click New . Select the template and click Add to create the selected template .
Baseline + ReWE + ReSE:	Press the File tab to access the chart view ; press New . Select the template and click Create to use the selected template .

Example 2:	
Src:	Na tomto projektu bylo skvělé , že žáci viděli lokální problém a bum – okamžitě se s ním snaží vyrovnat .
Ref:	What was really cool about this project was that the students saw a local problem , and boom – they are trying to immediately address it .
Baseline:	In this project , it was great that the kids had seen local problems and boom – immediately he’s trying to deal with him .
Baseline + ReWE:	In this project , it was great that the kids saw a local issue , and boom – they immediately try to deal with it .
Baseline + ReWE + ReSE:	What was great about this project was that the students saw a local problem, and boom , they’re trying to deal with him .

Table 5.7: Translation examples. Example 1: Eu-En and Example 2: Cs-En.

Chapter 6

Leveraging Discourse Rewards for Document-Level Neural Machine Translation

6.1 Introduction

As mentioned in previous chapters, the recent advances in neural machine translation (NMT) [Sutskever *et al.*, 2014; Bahdanau *et al.*, 2015; Luong *et al.*, 2015; Vaswani *et al.*, 2017] have provided the research community and the commercial landscape with effective translation models that can at times achieve near-human performance. However, this usually holds at phrase or sentence level. When using these models in larger units of text, such as paragraphs or documents, the quality of the translation may drop considerably in terms of discourse attributes such as lexical and stylistic consistency.

In fact, document-level translation is still a very open and challenging problem. The sentences that make up a document are not unrelated pieces of text that can be predicted independently; rather, a set of sequences linked together by complex underlying linguistics aspects, also known as the discourse [Maruf *et al.*, 2019b; Jurafsky and Martin, 2019]. The discourse of a document includes several properties such as grammatical cohesion [Halliday and Hasan, 2014], lexical cohesion [Halliday and Hasan, 2014], document coherence [Hobbs, 1979] and the use of discourse connectives [Kalajahi *et al.*, 2012]. Ensuring that the translation retains such linguistic properties is expected to significantly improve its overall readability and flow.

However, due to the limitations of current decoder technology, NMT models are still bound to translate at sentence level. In order to capture the discourse properties of the source document in the translation, researchers have attempted to incorporate more contextual information from surrounding sentences. Most document-level NMT approaches augment the model with multiple encoders, extra attention layers and memory caches to encode the surrounding sentences, and leave the model to implicitly learn the discourse attributes by simply minimizing a conventional NLL objective. The hope is that the model will spontaneously identify and retain the discourse patterns within the source document. Conversely, very little work has attempted to model the discourse attributes explicitly. Even the evaluation metrics typically used in translation such as BLEU [Papineni *et al.*, 2002] are not designed to assess the discourse quality of the translated documents.

Another problem to train document-level translation models is the limited supervised training data available. Most corpora publicly available on the Internet are at sentence-level [Bojar *et al.*, 2016; Bojar *et al.*, 2017; Bojar *et al.*, 2018], are completely shuffled and do not contain document boundaries. On the other hand, the available document-level datasets are usually small, with little number of documents. This makes document-level translation a low-resource NLP task, and thus, it is understandable that current NMT approaches struggle to learn the complex discourse structure of the documents.

For these reasons, in this chapter we propose to train an NMT model by directly targeting two specific discourse metrics: *lexical cohesion* (LC) and *coherence* (COH). LC is a measure of the frequency of semantically-similar words co-occurring in a document (or block of sentences) [Halliday and Hasan, 2014]. For example, *car*, *vehicle*, *engine* or *wheels* are all semantically-related terms. There is significant empirical evidence that ensuring lexical cohesion in a text eases its understanding [Halliday and Hasan, 2014]. At its turn, COH measures how well adjacent sentences in a text are linked to each other. In the following example from Hobbs [1979]:

“*John took a train from Paris to Istanbul. He likes spinach.*”

the two sentences make little ‘sense’ one after another. An incoherent text, even if grammatically and syntactically perfect, is anecdotally very difficult to understand and therefore coherence should be actively pursued. Relevant to translation, Vasconcellos [1989] has found that a high percentage of the human post-editing changes over machine-generated translations involves the improvement of cohesion and coherence.

Several LC and COH metrics that well correlate with the human judgement have been proposed in the literature. However, like BLEU and most other evaluation metrics, they are discrete, non-differentiable functions of the model's parameters. Hereafter, we propose to overcome this limitation by using the well-established *policy gradient* approach from reinforcement learning [Sutton and Barto, 2018] which allows using any evaluation metric as a reward without having to differentiate it. By combining different types of rewards, the model can be trained to simultaneously achieve more lexically-cohesive and more coherent document translations, while at the same time retaining faithfulness to the reference translation.

The rest of the chapter is organized as follows. Section 6.2 discusses related work. Section 6.3 describes the baseline NMT architectures used for the experiments. Section 6.4 presents the proposed training approach and the discourse rewards used with it. Section 6.5 presents the experiments and, finally, Section 6.6 concludes the paper.

6.2 Related Work

6.2.1 Document-level NMT

Many document-level NMT models have proposed taking the context into account by concatenating surrounding sentences or extra features to the current input sentence, with otherwise no modifications to the model. For example, Rios et al. [2017] have trained an NMT model that learns to disambiguate words given the context semantic landscape by simply extracting lexical chains from the source document, and using them as additional features. Other researchers have proposed concatenating previous source and target sentences to the current source sentence, so that the decoder can observe a proper amount of context [Agrawal *et al.*, 2018; Tiedemann and Scherrer, 2017; Scherrer *et al.*, 2019]. Their work has shown that concatenating even just one or two previous sentences can result in a noticeable improvement. Macé and Servan [2019] have added an embedding of the entire document to the input, and shown promising results in English-French.

Conversely, other document-level NMT approaches have proposed modifications to the standard encoder-decoder architecture to more effectively account for the context from surrounding sentences. Jean et al. [2017] have introduced a dedicated attention mechanism for the previous source sentences. Multi-encoder approaches with hierarchical attention networks have been proposed to separately encode each of the context sentences before they are merged back into a single context vector in the decoder [Miculicich *et al.*, 2018; Maruf *et al.*, 2019a; Wang *et al.*, 2017]. These models have shown significant improvements over sentence-level NMT baselines on many different language pairs. Kuang et al. [2018] and Tu et al. [2018] have proposed using an external cache to store, respectively, a set of topical words or a set of previous hidden vectors. This information has proved to benefit the decoding step at limited additional computational cost. In turn, Maruf and Haffari [2018] have presented a model that incorporates two memory networks, one for the source and one for the target, to capture document-level interdependencies. For the inference stage, they have proposed an iterative decoding algorithm that incrementally refines the predicted translation.

However, all the aforementioned models assume that the model can implicitly learn the occurring discourse patterns. Moreover, the training objective is the standard negative log-likelihood (NLL) loss, which simply maximizes the probability of the reference target words in the sentence. Only one work these authors are aware of ([Xiong *et al.*, 2019]) has attempted to train the model by explicitly learning discourse attributes. Inspired by recent work in text generation [Bosselut *et al.*, 2018], Xiong et al. [2019] have proposed automatically learning neural rewards that can encourage translation coherence at document level. However, it is not clear whether the learned rewards would be in good correspondence with human judgment. For this reason, in our work we prefer to rely on established discourse metrics as rewards. Finally, Tebbifakhr et al. [2019] have used a similar reinforcement learning based loss for a different task. Their work has used a text sentiment scorer to generate translations that improve a downstream sentiment classifier predict whether a tweet has positive or negative sentiment. They have shown that their training improves the accuracy of the downstream task classifier in comparison to translations generated by a general-purpose NMT model.

6.2.2 Discourse evaluation metrics

As a matter of fact, several metrics have been proposed in the literature to measure discourse properties. Early work of Morris and Hirst [1991] have proposed to model document's lexical cohesion by computing lexical chains using a thesaurus as a major knowledge base. Wong and Kit [2012] have suggested an alternative metric that looks for repetitions of words and their related terms (e.g. hyponyms, hypernyms) by using WordNet [Miller, 1998]. Their work was mainly proposed as additional metric to evaluate the quality of the output of MT systems at document-level. The proposed LC metric have shown strong Pearson's correlation with human translations. Later, Gong et al. [2015] have proposed a similar metric based on the computation of lexical chains. For COH, two types of metrics have been proposed: *entity-based* and *topic-based*. The former follow the Centering Theory [Grosz *et al.*, 1995] which states that documents with a high frequency of the same salient entities are more coherent. An entity-based coherence metric was proposed by Barzilai and Lapata [2008]. At their turn, *topic-based* metrics assume that a document is coherent when adjacent sentences are similar in topic and vocabulary. Accordingly, Hearst [1997] has proposed the Textilling algorithm which computes the cosine distance between the bag-of-word (BoW) vectors of adjacent sentences. Foltz et al. [1998] have proposed to replace the BoW vectors with topic vectors. They have applied this metric in the coherence analysis of several texts [Britton and Gülgöz, 1991; McNamara *et al.*, 1996], discourse segmentation and as a measurement of semantic distance, and have showed better correlation with human judgment. Li et al. [2017] have learned topic embeddings with a self-supervised neural network. Other metrics have been proposed to measure different discourse properties such as grammatical cohesion [Hardmeier and Federico, 2010; Miculicich and Popescu-Belis, 2017] and discourse connectives [Hajlaoui and Popescu-Belis, 2013].

6.2.3 Reinforcement learning in NMT

Finally, researchers in NMT and other natural language generation tasks have used reinforcement learning [Sutton and Barto, 2018] techniques to train the models to maximize discrete sentence-level and document-level metrics as an alternative or a complement to the NLL. For example, Ranzato et al. [2015] have proposed training NMT systems targeting the BLEU score, showing consistent improvements with respect to strong baselines. In addition to training the model directly with the evaluation function, they claim that this approach mollifies the exposure bias problem [Bengio *et al.*, 2015]. Shen et al. [2016] have proposed a similar solution using minimum risk training. Paulus et al. [2018] have proposed a similar approach for summarization using ROUGE as the training loss [Lin and Hovy, 2000]. Finally, Edunov et al. [2018b] have presented a comprehensive comparison of reinforcement learning and structured prediction losses for NMT model training.

6.3 Baseline Models

This section describes the baseline NMT models used in the experiments. In detail, subsection 6.3.1 recaps the standard sentence-level translation model while subsection 6.3.2 describes the recent, strong hierarchical baseline that we augment with discourse rewards.

6.3.1 Sentence-level NMT

Our first baseline is a standard sentence-level NMT model. Given the source document D with k sentences, the model translates each sentence $\mathbf{x}_i = \{\mathbf{x}_i^1, \ldots, \mathbf{x}_i^{n_i}\}, i = 1, \ldots, k$, in the document into a sentence in the target language, $\mathbf{y}_i^\star = \{\mathbf{y}_i^1, \ldots, \mathbf{y}_i^{m_i}\}$:

$$\mathbf{y}_i^\star = \arg\max_{\mathbf{y}_i} p(\mathbf{y}_i|\mathbf{x}_i, \theta) \quad i = 1, \ldots, k \tag{6.1}$$

Thus, the model translates every sentence in the document independently. Our sentence model uses a standard transformer-based encoder-decoder architecture [Vaswani *et al.*, 2017] where the model is trained to maximize the probability of the words in the training reference sentences using an NLL objective. We train this model for 20 epochs and select the best model over the validation set.

6.3.2 Hierarchical Attention Network

As a document-level translation baseline, we have used the Hierarchical Attention Network (HAN) of Miculicich et al. [2018]. A HAN network is added to the sentence-level

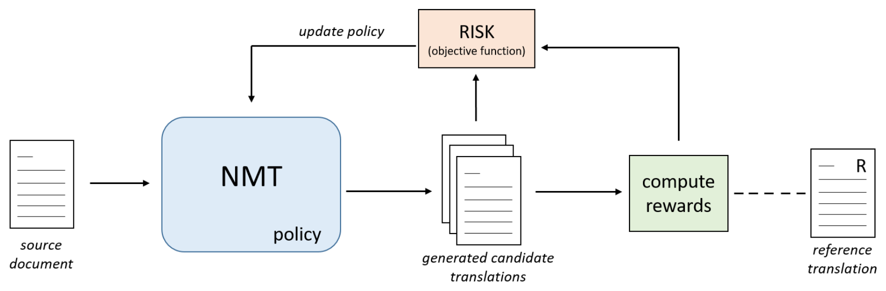


Figure 6.1: RISK training. Given the source document, the policy (NMT model) predicts l candidate translations. Then, a reward function is computed for each such translation. For supervised rewards, (e.g., BLEU) the reference translation is required, but not for LC and COH. Finally, the RISK loss is computed using the rewards and the probabilities of the candidate translations, differentiated, and backpropagated for parameter update.

NMT model both in the encoder and in the decoder (referred to as HAN$_{\text{join}}$ in the following), allowing the model to encode information from t previous source and target sentences. The prediction can be expressed as:

$$\mathbf{y}_i^\star = \arg\max_{\mathbf{y}_i} p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{x}_{i-1}, \ldots, \mathbf{x}_{i-t}, \mathbf{y}_{i-1}, \ldots, \mathbf{y}_{i-t}, \theta) \quad i = 1, \ldots, k \tag{6.2}$$

where $(\mathbf{x}_{i-1}, \ldots, \mathbf{x}_{i-t})$ are the previous source sentences and $(\mathbf{y}_{i-1}, \ldots, \mathbf{y}_{i-t})$ the previous target sentences that make up the context. At inference time, the target sentences are the model's own predictions. Following the indications given by the authors, we have set $t = 3$. Additionally, we have used the weights of the sentence-level NMT baseline to initialize the common parameters of the HAN$_{\text{join}}$ model, and we have initialized the extra parameters introduced by the HAN networks randomly. The model has been fine-tuned for 10 epochs and the best model over the validation set has been selected.

6.4 RISK training with discourse rewards

In order to improve the baseline models, we propose to use the LC [Wong and Kit, 2012] and COH [Foltz *et al.*, 1998] evaluation metrics as rewards during training, so that the model is explicitly rewarded for generating more cohesive and coherent translation at document level. For that, we use a reinforcement learning approach, which allows using discrete, non-differentiable functions as rewards in the objective. Following Edunov et al. [2018b], we have used the structured loss that achieved the best results in their experiments, namely the *expected risk minimization (RISK)* objective:

$$\mathcal{L}_{RISK} = \sum_{\mathbf{u} \in \mathcal{U}(\mathbf{x})} -r(\mathbf{u}, \mathbf{y}) p(\mathbf{u}|\mathbf{x}, \theta) \tag{6.3}$$

where **x** is the source sentence, **y** is the reference translation, $p(\mathbf{u}|\mathbf{x})$ is the conditional probability of a translation in our 'policy', or NMT model, $\mathcal{U}(x)$ is a set of candidate translations generated by the current policy, and $r(\cdot)$ is the reward function. In our work we have obtained the candidate translations using beam search, which achieved higher accuracy than sampling in Edunov et al. [2018b]. The conditional probability of a translation is computed as:

$$\begin{aligned} p(\mathbf{u}|\mathbf{x}, \theta) &= \frac{f(\mathbf{u}, \mathbf{x}, \theta)}{\sum_{\mathbf{u'} \in \mathcal{U}(\mathbf{x})} f(\mathbf{u'}, \mathbf{x}, \theta)} \\ f(\mathbf{u}, \mathbf{x}, \theta) &= \exp[\frac{1}{m} \sum_{j=1}^{m} \log p(u^j | u^1, \ldots, u^{j-1}, \mathbf{x}, \theta)] \end{aligned} \tag{6.4}$$

where m is the number of words in the candidate translation. Note that in order to avoid underflow and put all sentences in a similar scale, the conditional probability is computed as a sum of logarithms, divided by the number of tokens in the sequence and, finally, brought back to probability scale with the exponential function.

By minimizing this RISK objective, the NMT model is encouraged to give higher probability to candidate translations that obtain a higher reward. This function has been used at sentence level by Edunov et al. [2018b]. However, the same metrics could also be computed at document level by simply concatenating all the sentences from the same document together (both for the ground truth and the predictions). As a result, m now would be the number of words in a document, $\mathcal{U}(\mathbf{x})$ the candidate document translations, **x** the source document and **y** the reference document. Computing the RISK objective in this way permits having document-level reward functions as $r(\cdot)$.

6.4.1 Reward functions

We have explored the use and combination of different reward functions for training:

LC$_{\text{doc}}$: For LC, the metric proposed by Wong and Kit [2012] has been adopted. This metrics counts the number of lexical cohesive devices in the document and then divides that number by the total number of words in the document (Eq. 6.5). Cohesive devices include associations such as repetitions of words, synonyms, near-synonyms, hypernyms, meronyms, troponyms, antonyms, coordinating terms, and so on. WordNet [Fellbaum, 2012] has been used to classify the relationships between words. Note that this reward function is unsupervised since it does not require the ground-truth reference to compute it.

$$LC = \frac{\#\ of\ cohesive\ devices\ in\ document}{\#\ of\ words\ in\ document} \tag{6.5}$$

COH$_{\text{doc}}$: To calculate COH, we have used the approach proposed by Foltz et al. [1998]. This approach first uses a trained LSA model to infer topic vectors ($\mathbf{t}_i$) for each sentence in the document, and then computes the average cosine distance between adjacent sentences (Eq. 6.6). For the topic vectors, we have used the pre-trained LSA model (Wiki-6) from Stefanescu et al. [2014], which was trained over Wikipedia. Note that COH also does not require a ground-truth reference.

$$COH = \frac{1}{k-1}\sum_{i=2}^{k}\cos(\mathbf{t}_i, \mathbf{t}_{i-1}) \tag{6.6}$$

BLEU$_{\text{doc}}$: In addition to the LC and COH rewards, we have decided to use a reference-based metric such as BLEU [Papineni *et al.*, 2002]. Due to the unsupervised nature of LC and COH, the model could trivially boost them by only repeating words and creating very similar sentences. However, this will come at the expense of producing translations that are increasingly unrelated to the reference translation (low adequacy) and grammatically incorrect (low fluency). As such, we encourage the model to also target a high BLEU score in its predictions.

BLUE$_{\text{sen}}$: Finally, we have also used BLEU at sentence level as a reward. In this way, we can assess whether it is more beneficial to use this metric at document or sentence level.

Language pair	Domain	train	dev	test	Avg. # sent/doc
Zh-En	TEDtalks	0.2M	0.9K	3.9K	122
Cs-En	TEDtalks	0.1M	0.5K	5.2K	114
Eu-En	subtitles	0.8M	0.8K	1.5K	1018
Es-En	subtitles	1.1M	1.9K	4.6K	774
Es-En	news	0.2M	2.1K	14K	37

Table 6.1: The datasets used for the experiments.

These four rewards can be combined in several different ways. To limit the experiments, we have decided to use them in their natural range without reweighting. All the results with the different reward combinations are presented in Section 6.5.2.

6.4.2 Mixed objective

Similar to the MIXER training proposed by Ranzato et al. [2015], we have also explored mixing the RISK objective with the NLL. The rationale is similar to that of using BLEU_{doc} and BLEU_{sen} as rewards: the NLL loss can help the model not to deviate too much from the reference translation while improving discourse properties. To mix these losses, we have used an alternate batch approach: either loss is randomly selected in each training batch, with a certain probability (e.g. RISK(0.8) means that we have selected the RISK loss with 80% probability and the NLL with 20%).

6.5 Experiments

6.5.1 Datasets and experimental setup

We have performed a broad range of experiments over four different language pairs and three different translation domains (TED talks, movie subtitles and news) which have been used by other popular document level NMT research [Miculicich *et al.*, 2018; Tu *et al.*, 2018]. In this way we aim to facilitate the comparison of our research with the existing literature, and limit the size of the training datasets required to test the proposed training loss in a low-resource scenario. For translations of TED talks, we have used the datasets released in the IWSLT15[1] shared task for Chinese-English (Zh-En) and in

[1]https://sites.google.com/site/iwsltevaluation2015/mt-track

IWSLT16[2] for Czech-English (Cs-En). TED talks are usually 20 minute talks that focus around a main topic. Thus, we believe that rewards that encourage lexical cohesion and topic coherence among the document will help the system to understand the context and correctly disambiguate the meaning of many words in the document. For both language pairs, we have used their *dev2010* set as the validation set, and sets *tst2011-2013* (Zh-En) and *tst2010-2013* (Cs-En), respectively, as test sets. For translations in the movie subtitles domain, we have used the OpenSubtitles-v2018 dataset [Lison *et al.*, 2018] from OPUS[3], and the language pairs tested have been Basque-English (Eu-En) and Spanish-English (Es-En). Movie subtitles are usually conversations between the characters in the movie, where they may be discussions, interruptions, monologues etc. For that reason, we may expect a lower coherence between adjacent sentences, yet tracking lexical cohesion is probably still a strong feature in the documents. For Eu-En we have used all the available data, but for Es-En we have only used a subset of the corpus to limit time and memory requirements. In both cases, we have divided the data into a training, validation and test sets[4]. The last translation domain is news, for which we have used the Es-En News-Commentary11 dataset[5]. Similar to the TED talks, these are focused on a single topic, and usually are shorter documents, written in formal language. As validation and test sets, we have used its *newstest2008* and *newstest2009-2013* sets, respectively, from WMT[6]. All the datasets have been tokenized using the *Moses tokenizer*[7], with the exception of Chinese for which we have used *Jieba*[8]. Sentences with more than 50 tokens have been excluded. A *truecaser* model from moses[7] has been learned over the training data of each dataset, and has been applied for consistent word casing as a final pre-processing step.

For training of the sentence-level baseline model, we have used the hyper-parameters proposed by Miculicich et al. [2018] in their code repository[9]. The model uses a 6-layer transformer network [Vaswani *et al.*, 2017] as the encoder and decoder. The dimension of the source word embeddings, the target word embeddings and the transformers' hidden

[2]https://sites.google.com/site/iwsltevaluation2016/mt-track
[3]http://opus.nlpl.eu/
[4]All the datasets will be released publicly, and the reviewers can already see them as supplementary material.
[5]http://www.casmacat.eu/corpus/news-commentary.html
[6]http://www.statmt.org/wmt13/translation-task.html
[7]https://github.com/moses-smt/mosesdecoder
[8]https://github.com/fxsjy/jieba
[9]https://github.com/idiap/HAN_NMT

vectors have all been set to 512. The default position encoding was added to the input vectors and a dropout of 0.1 to the hidden vectors. Additionally, label smoothing of 0.1 was applied to the output probabilities. During training, the batch size was set to 4096 tokens with a gradient accumulation of 4. We have used the Adam optimizer [Kingma and Ba, 2015] with a learning rate of 2, and $\beta_2 = 0.998$. The parameters of the network have been initialized with the *glorot* method [Glorot and Bengio, 2010], and the model has been warmed up with $8,000$ steps. It has been trained for 20 epochs and the model with best perplexity over the validation set has been selected.

The document-level HAN_{join} baseline follows almost exactly the settings of the sentence-level one. The main difference is the added HAN networks in the encoder and the decoder. During training, for memory reasons the batch size has been reduced to $1,024$ tokens, and the learning rate to 0.2. The parameters in common have been initialized with the pre-trained sentence-level baseline, while the extra HAN networks have been initialized with *glorot*. For computational reasons, this model has been trained for only 10 epochs. In the validation, the best model was selected as that with the highest values in the majority of the evaluation metrics (BLEU, LC, COH and F_{BERT}).

For our proposed training approach, we have compared multiple models trained with the RISK objective with different combinations of reward functions. This has allowed us to select the best reward functions for the translation quality at document level. Then, the model trained with the best reward combination has been compared against the sentence-level NMT and HAN baselines. In our experiments, the RISK training objective has been used as fine-tuning of a pre-trained HAN_{join} baseline model, in order not to suffer from a “cold start” due to the large output label class. The main aim of our experiments is to show that the proposed training objectives can lead to performance improvements over HAN_{join}. Candidate translations have been obtained using beam search with a beam size of only 2, due to memory and computational time limitations. Furthermore, the training batch size has been set to 15 sentences. Since the objective is computed over the batch, this is equivalent to subdividing longer documents into sub-documents of 15 sentences each. Yet, our experimental results show that computing the rewards at such batch level is still effective for improving the translation quality. The models have been fine-tuned until convergence of the perplexity on the validation set, and using simulated annealing

[Denkowski and Neubig, 2017], which repeatedly halves the learning rate when perplexity convergence is reached. The number of annealing steps was set to 5.

Each of the baselines and proposed models have been trained with three different seeds over its training set, and the validation set has been used at all times to select the best model. Then, the average results of the three runs over the test sets have been reported. We have measured four different evaluation metrics: BLEU, LC, COH and F_{BERT}, an alternative metric to BLEU that compares the BERT sentence embeddings of the prediction and the reference and which has been shown to have better correlation with the human judgement than BLEU [Zhang *et al.*, 2020]. To select the best model over the validation set for the sentence-level NMT baseline, we have used the lowest perplexity. Instead, for the HAN$_{\text{join}}$ baseline and our models, we have chosen the model with the best results in the majority of the four evaluation metrics (BLEU, LC, COH and F_{BERT}). This has not altered the relative ranking of the sentence-level NMT baseline since it has performed the worst in all cases anyway.

6.5.2 Results

Table 6.2 shows the main results from our experiments. Over all datasets, the HAN$_{\text{join}}$ baseline has consistently outperformed the sentence-level NMT in terms of BLEU score and F_{BERT} which shows that including surrounding sentences can help to obtain better translation accuracy. However, HAN$_{\text{join}}$ has not performed significantly better than the sentence-level model in terms of LC and COH (even worse in a few cases), showing that it has not been able to specifically learn the discourse properties in the document.

Table 6.2 also shows the results from our best models in comparison to these baselines. From preliminary experiments, we have seen that the RISK model that achieved the best results is the one that combines BLEU$_{\text{doc}}$, LC$_{\text{doc}}$ and COH$_{\text{doc}}$ as rewards. Yet, choosing the right proportion of RISK and NLL training has proven very important and dataset-dependent. In the TED talks domain (Table 6.2a), we can see that the proposed training approach has been able to improve the LC, COH and F_{BERT} metrics over both baselines. The Zh-En model with RISK(1.0) has outperformed the HAN$_{\text{join}}$ baseline by 2.46 percentage points (pp) in LC, 1.17 pp in COH and 0.48 pp in F_{BERT}, while the Cs-En model has improved by 2.68 pp, 0.55 pp and 0.22 pp, respectively. For this dataset, the

Model	**Zh-En (TED talks)**				**Cs-En (TED talks)**			
	BLEU	LC	COH	F_{BERT}	BLEU	LC	COH	F_{BERT}
Sentence-level NMT	16.94	55.39	28.02	66.94	22.74	55.62	27.72	69.60
HAN$_{\text{join}}$	17.52	55.02	28.15	67.21	23.44	55.63	27.62	69.87
RISK(1.0)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	**18.15**	**57.48***	**29.32***	67.69	23.40	**58.31***	**28.17**	**70.09**
RISK(0.8)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	17.82	55.18	28.68	67.60	23.43	56.03*	27.62	70.01*
RISK(0.5)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	17.83	54.70	28.30	**67.73**	23.42	56.07	27.78	69.95*
RISK(0.2)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	17.80	55.10	28.35	67.62	**23.48**	55.85	27.62	69.95
Human reference	–	55.13	29.33	–	–	55.91	29.7	–

(a) Results over the TED talks datasets.

Model	**Eu-En (movie subtitles)**				**Es-En (movie subtitles)**			
	BLEU	LC	COH	F_{BERT}	BLEU	LC	COH	F_{BERT}
Sentence-level NMT	9.12	37.08	19.34	59.18	29.34	58.31	22.70	67.57
HAN$_{\text{join}}$	9.74	37.19	19.63	59.72	**30.14**	58.11	22.58	**67.73**
RISK(1.0)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	1.19	*72.51**	*27.67*	36.72	3.37	*67.82*	19.53	48.07
RISK(0.8)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	9.67	40.66*	19.60	**59.76**	29.51	58.34	22.82	67.51
RISK(0.5)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	9.77	38.85*	19.80	59.62	29.79	**58.44**	22.76	67.53
RISK(0.2)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	**9.99**	37.53	19.42	59.72	29.70	58.39	**22.96**	67.50
Human reference	–	41.83	21.93	–	–	57.28	24	–

(b) Results over the movie subtitles datasets.

Model	**Es-En (news)**			
	BLEU	LC	COH	F_{BERT}
Sentence-level NMT	21.79	32.97	28.10	67.88
HAN$_{\text{join}}$	22.16	32.87	28.15	**68.28**
RISK(1.0)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	20.67	32.81	28.14	67.84
RISK(0.8)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	22.26	**33.70***	**28.45***	68.14
RISK(0.5)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	22.34	33.51*	28.39	68.02
RISK(0.2)-$BLEU_{doc} + LC_{doc} + COH_{doc}$	**22.45***	33.32*	28.25	68.13
Human reference	–	38.66	30.97	–

(c) Results over the news datasets.

Table 6.2: Main results. (*) means that the differences are statistically significant with respect to the HAN$_{\text{join}}$ baseline with a p-value < 0.05 over a one-tailed Welch's t-test. LC and COH values that come at the expense of a drop in translation accuracy (e.g. BLEU, F_{BERT}) are highlighted in italics.

best performance has been achieved with RISK(1.0) (pure RISK training, no NLL). In the case of Zh-En, our best model has also significantly improved the BLEU score, while in the case of Cs-En the BLEU scores of the HAN$_{\text{join}}$ baseline and our models have been roughly equivalent. In general, we had not anticipated the improvements in BLEU score and F_{BERT} since our main aim had only been to improve the translations in terms of discourse metrics. However, in some cases the improvements in discourse metrics have also translated in higher translation accuracy.

In turn, Table 6.2b shows the main results over the movie subtitles datasets which are characterized by documents with on average more, yet much shorter, sentences than the

Model	BLEU	LC	COH	F_{BERT}
$BLEU_{doc} + LC_{doc} + COH_{doc}$	**18.15**	57.48	29.32	67.69
$BLEU_{sen} + LC_{doc} + COH_{doc}$	17.53	56.32	28.79	**67.96**
$BLEU_{doc} + LC_{doc}$	17.44	**59.21**	**29.87**	67.27
$BLEU_{doc} + COH_{doc}$	17.57	55.74	28.82	67.41
$BLEU_{sen} + LC_{doc}$	17.60	56.32	28.76	67.87
$BLEU_{sen} + COH_{doc}$	17.46	56.31	28.82	67.93
$LC_{doc} + COH_{doc}$	10.56	*71.28*	*31.25*	62.27
$BLEU_{sen}$	17.42	55.93	28.76	67.83
$BLEU_{\text{doc}}$	17.20	54.59	28.15	67.18
LC_{doc}	10.42	*71.70*	*31.61*	62.09
COH_{doc}	17.26	58.66	29.98	66.92

Table 6.3: Ablation study of the various reward functions over the Zh-En TED talks dataset with RISK(1.0). Undesirable LC and COH values are highlighted in italics.

TED talks. On these datasets, the RISK(1.0) model has been able to improve the LC and COH metrics to a large extent, but at a marked cost in BLEU score and F_{BERT}. The translations generated by this model have often displayed many word and phrase repetitions that had little correspondence with the reference translation. This shows that improving the LC and COH metrics beyond a certain extent is undesirable. Conversely, training the model with the mixed objective has forced it to stay closer to the reference translations and helped it achieve higher BLEU and F_{BERT} scores. On Eu-En, the RISK(0.8) model has improved the LC by 3.47 pp at a substantial parity of all the other metrics. On Es-En, none of the proposed models has clearly outperformed the HAN$_{\text{join}}$ baseline. For instance, the RISK(0.5) model has improved LC and COH by 0.33 pp and 0.18 pp, respectively, but at the cost of 0.35 pp in BLEU score and 0.20 pp in F_{BERT}.

The proposed models have achieved better results on the news domain dataset (Table 6.2c) where they have been able to improve the BLEU score, LC and COH at a mild cost in F_{BERT}. In general, we can argue that the discourse rewards have been more effective on documents such as the TED talks, which come from single authors and are generally controlled in style, than on documents such as subtitles and news which are more fragmented in nature.

6.5.2.1 Ablation study

To expand the analysis, Table 6.3 shows the results from an ablation study conducted over the Zh-En dataset that explores the impact of the various reward functions. The best trade-off over the four evaluation metrics seems that returned by BLEU$_{\text{doc}}$ + LC$_{\text{doc}}$ + COH$_{\text{doc}}$ which has achieved the highest BLEU score, a high F_{BERT}, and high LC and COH. The results also show that using BLEU$_{\text{sen}}$ as a reward has contributed to improve the F_{BERT} score in all cases, but at the significant expense of the other evaluation metrics. However, when BLEU$_{\text{doc}}$ and BLEU$_{\text{sen}}$ have been compared head-to-head as the sole rewards, the sentence-level BLEU has been able to achieve higher scores in all metrics. In contrast, the BLEU$_{\text{doc}}$ reward has been most effective when used jointly with the cohesion and coherence rewards. At its turn, the LC$_{\text{doc}}$ reward without a balance from a BLEU reward has led to LC and COH scores that are likely excessive and undesirable, with a corresponding drop in BLEU score and F_{BERT}. Conversely, the COH$_{\text{doc}}$ reward has not displayed a comparable degradation. The main overall result from this ablation analysis is that the rewards need to be used in a calibrated combination to deliver the best trade-off across all the evaluation metrics, and that the selection of the best combination can be effectively carried out by validation.

Moreover, we have also tracked the improvement of the targeted evaluation metrics (i.e. BLEU, LC and COH) during training. Figure 6.2 shows the BLEU, LC and COH scores over the Cs-En validation set at different training iterations. This plot reinforces the idea that improving LC and COH comes at a cost of BLEU score. In the first 2000 training iterations, the LC has improved by more than 2 p.p. and the coherence by more than 1 p.p., while the BLEU score has dropped by less than 0.5. Additionally, the highest score of LC and COH matches with the lowest BLEU score (iteration 4000). Overall, we have concluded that the validation set plays a key role for early stopping of the training and to achieve a model with the right balance of BLEU, LC and COH (e.g. iteration 2000 or 6800).

6.5.2.2 Translation examples

Finally, Tables 6.4 and 6.5 show two examples of document-level translations made by our models, in comparison to the reference translation (Ref) and the HAN$_{\text{join}}$ baseline's, given the sentence in the source language (Src). The first example is a snippet of a document in the Zh-En TED talks dataset. Our best model on this dataset has been that trained with RISK(1.0) and BLEU$_{\text{doc}}$, LC$_{\text{doc}}$ and COH$_{\text{doc}}$ as rewards. In this example, we

Src:	. . . 女士们，先生们，见见你的近亲。 这就是野生倭黑猩猩的世界座落于刚果的丛林中 。 倭黑猩猩 和黑猩猩是我们大家生活里最密切相关的近亲。 这意味我们都享有一个共同的祖先，一个进化了的祖母，她生活在大约6 百万年前 。 . . .
Ref:	. . . ladies and gentlemen , meet your cousins . this is the world of wild bonobos in the jungles of Congo . bonobos are , together with chimpanzees , your living closest relative . that means we all share a common ancestor , an evolutionary grandmother , who lived around six million years ago . . .
HAN$_{\text{join}}$:	. . . ladies and gentlemen , meet your relatives . this is the world of the wildlife that is in the Congo . the chimps and chimpanzees are the most closely related to us . it means we all have a common ancestor , a grandmother who has evolved about six million years ago . . .
RISK(1.0):	. . . ladies and gentlemen , meet your close relatives . and that 's the world of the wild bonobos that are in the jungle in the Congo . the bonobos are the most closely related to the chimpanzees that we live in . it means that we all have a common ancestor , a grandmother who lived about six million years ago . . .

Table 6.4: Translation example. Snippet of a document from the Zh-En TED talks test set.

Src:	. . . no voy a perdonar a ese bastardo ! Digaselo al Dr Chaddha , no me mienta . le dí el 30 % por adelantado . . . incluso después de haberme prometido , que él nos daria una esperma de calidad . digale que se joda !. . .
Ref:	. . . I wont spare that bas**** tell that Dr. Chaddha of yours , not to lie to me . he has taken 30 % advance from me . . . even after promising , he hasn 't given us a quality sperm . you tell that f * * * er I 'll hunt him down. . .
HAN$_{\text{join}}$:	. . . I 'm not going to forgive that bastard ! don 't lie to me . I gave him 30 % earlier . even after I was promised , he 'd give us a quality sperm. . .
RISK(0.8):	. . . I 'm not going to forgive that bastard ! don 't lie to me . I gave him 30 % advance . even after I was promised , he 'd give us a quality sperm . . .

Table 6.5: Translation example. Snippet of a document from the Es-En subtitles test set.

have observed clearly the positive influence of the LC and COH rewards, as the model has been able to provide better lexical cohesion and coherence in the translation. The model has been able to correctly translate words such as *bonobos* and *jungle* while the HAN$_{\text{join}}$ model has uttered a more generic *chimps*. In addition, the translation generated by our model seems more faithful to the reference. In the second example, our best model (RISK(0.8)+BLEU$_{\text{doc}}$+ LC$_{\text{doc}}$ + COH$_{\text{doc}}$) seems to have better captured the context of the snippet, which revolves around money and payments, and has correctly translated the Spanish word *adelanto* for *advancement*. Conversely, the translation from the HAN$_{\text{join}}$ baseline has been *earlier*, which could be correct in a different context, but not in this.

6.6 Conclusion

In this chapter, we have presented a novel training method for document-level NMT models that uses discourse rewards to encourage the models to generate more lexically cohesive and coherent translations at document level. As training objective we have used a reinforcement learning-style function, named RISK, that permits using discrete, non-differentiable terms in the objective. Our results on four different language pairs and three translation domains have shown that our models have achieved a consistent improvement in discourse metrics such as LC and COH, while retaining comparable values of accuracy metrics such as BLEU and F_{BERT}. In fact, on certain datasets, the models have even improved on those metrics. While the approach has proved effective in most cases, the best combination of discourse rewards, accuracy rewards and NLL has had to be selected by validation for each dataset. In the near future we plan to investigate how to automate this selection, and also explore the applicability of the proposed approach to other natural language generation tasks.

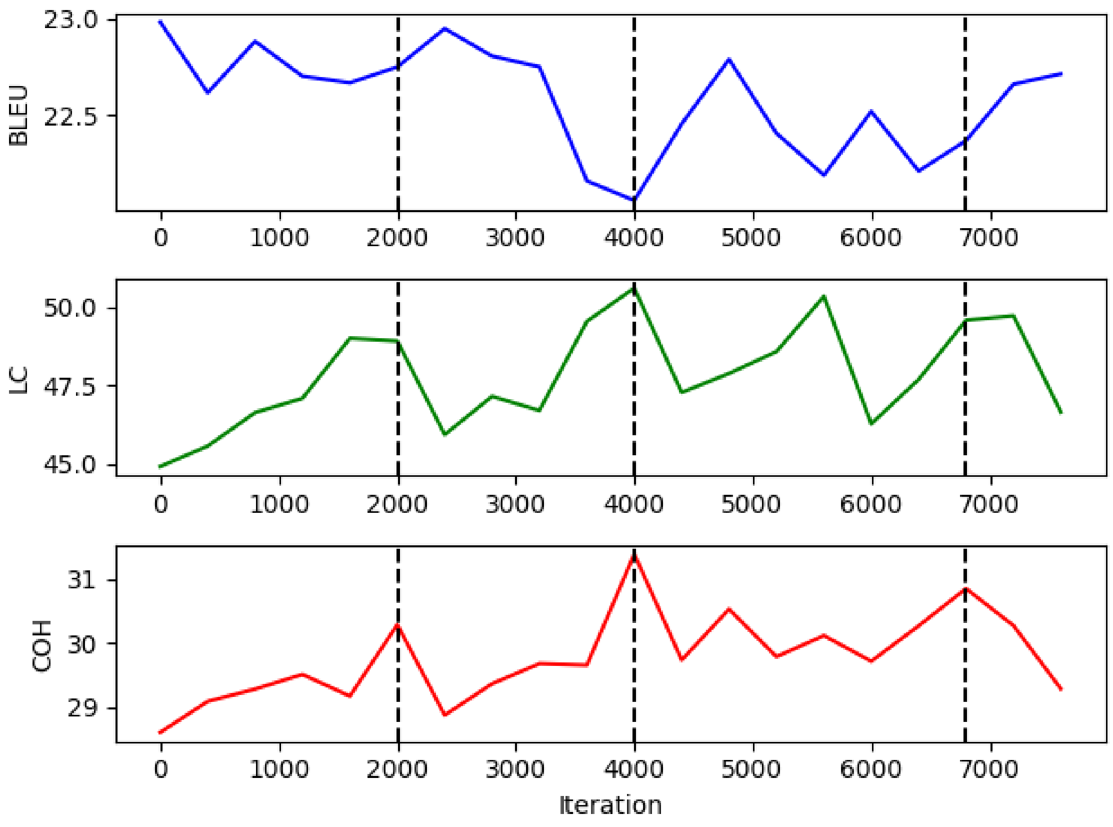


Figure 6.2: BLEU, LC and COH scores over the Cs-En validation set at different training iterations.

Chapter 7

A Shared Attention Mechanism for Interpretation of Neural Automatic Post-Editing Systems

7.1 Introduction

In current professional practice, translators tend to follow a two-step approach: first, they run a machine translator (MT) to obtain a first-cut translation; then, they manually correct the MT output to produce a result of adequate quality. The latter step is commonly known as *post-editing* (PE). Stemming from this two-step approach and the recent success of deep networks in MT [Sutskever *et al.*, 2014; Bahdanau *et al.*, 2015; Luong *et al.*, 2015], the MT research community has devoted increasing attention to the task of automatic post-editing (APE) [Bojar *et al.*, 2017].

The rationale of an APE system is to be able to automatically correct the systematic errors made by the MT and thus dispense with or reduce the work of the human post-editors. The data for training and evaluating these systems usually consist of triplets (*src*, *mt*, *pe*), where *src* is the sentence in the source language, *mt* is the output of the MT, and *pe* is the human post-edited sentence. Note that the *pe* is obtained by correcting the *mt*, and therefore these two sentences are closely related. An APE system is "monolingual" if it only uses the *mt* to predict the post-edits, or "contextual" if it uses both the *src* and the *mt* as inputs [Béchara *et al.*, 2011]. Due to the fact that APE systems focus on correcting the errors made by a general-purpose black-box MT system, they usually require less training data to be able to substantially improve the quality of the translations. Consequently, it is a suitable approach for low-resource scenarios.

Despite their remarkable progress in recent years, neural APE systems are still elusive when it comes to interpretability. In deep learning, highly interpretable models can help researchers to overcome outstanding issues such as learning from fewer annotations, learning with human-computer interactions and debugging network representations [Zhang and Zhu, 2018]. More specifically in APE, a system that provides insights on its decisions can help the human post-editor to understand the system's errors and consequently provide better corrections. As our main contribution, in this chapter we propose a contextual APE system based on the seq2seq model with attention which allows for inspecting the role of the *src* and the *mt* in the editing. We modify the basic model with two separate encoders for the *src* and the *mt*, but with a single attention mechanism shared by the hidden vectors of both encoders. At each decoding step, the shared attention has to decide whether to place more weight on the tokens from the *src* or the *mt*. In our experiments, we clearly observe that when the *mt* translation contains mistakes (word order, incorrect words), the model learns to shift the attention toward tokens in the source language, aiming to get extra "context" or information that will help to correctly edit the translation. Instead, if the *mt* sentence is correct, the model simply learns to pass it on word by word. In Section 7.4.4, we have plotted the attention weight matrices of several predictions to visualize this finding.

The model has been trained and evaluated with the official datasets from the WMT16 and WMT17 Information Technology (IT) domain APE English-German (en-de) shared tasks [Bojar *et al.*, 2016; Bojar *et al.*, 2017]. We have also used the 500K artificial data provided in the shared task for extra training. For some of the predictions in the test set, we have analysed the plots of attention weight matrices to shed light on whether the model relies more on the *src* or the *mt* at each time step. Moreover, our model has achieved higher accuracy than previous systems that used the same training setting (official datasets + 500K extra artificial data).

7.2 Related work

In an early work, [Simard *et al.*, 2007] combined a rule-based MT (RBMT) with a statistical MT (SMT) for monolingual post-editing. The reported results outperformed both systems in standalone translation mode. In 2011, [Béchara *et al.*, 2011] proposed the first model based on contextual post-editing, showing improvements over monolingual approaches.

More recently, neural APE systems have attracted much attention. [Junczys-Dowmunt and Grundkiewicz, 2016] (the winner of the WMT16 shared task) integrated various neural machine translation (NMT) components in a log-linear model. Moreover, they suggested creating artificial triplets from out-of-domain data to enlarge the training data, which led to a drastic improvement in PE accuracy. Assuming that post-editing is reversible, [Pal *et al.*, 2017] have proposed an attention mechanism over bidirectional models, *mt*$\rightarrow$ *pe* and *pe* $\rightarrow$ *mt*. The idea of using an attention mechanism has become very popular, and thus, several other researchers have proposed variations of this technique to use multi-input seq2seq models for contextual APE to allow the model to focus on specific sub-phrases of the inputs at each decoding step. [Bérard *et al.*, 2017; Libovický *et al.*, 2016; Varis and Bojar, 2017; Pal *et al.*, 2017; Libovický and Helcl, 2017; Chatterjee *et al.*, 2017]. All these systems employ separate encoders for the two inputs, *src* and *mt*. For instance, the winner of the 2017 shared task [Chatterjee *et al.*, 2017], proposed a system with two encoders, each with its own attention mechanism, with independent trainable parameters. Bérard et al. [2017] have proposed a similar approach, but their model predict edit operations instead of words in the target vocabulary. They have showed their attention mechanism works very well in low-resource scenarios.

7.2.1 Attention mechanisms for APE

A key aspect of neural APE systems is the attention mechanism. A conventional attention mechanism for NMT first learns the alignment scores (e^{ij}) with an alignment model [Bahdanau *et al.*, 2015; Luong *et al.*, 2015] given the j-th hidden vector of the encoder ($\mathbf{h}^j$) and the decoder's hidden state ($\mathbf{s}_{i-1}$) at time $i-1$ (Equation 7.1). Then, Equation 7.2 computes the normalized attention weights, with T_x the length of the input sentence. Finally, the context vector is computed as the sum of the encoder's hidden vectors weighed by the attention weights (Equation 7.3). The decoder uses the computed context vector to predict the output.

$$e^{ij} = aligment\_model(\mathbf{h}^j, \mathbf{s}^{i-1}) \tag{7.1}$$

$$\alpha^{ij} = \frac{exp(e^{ij})}{\sum_{m=1}^{T_x} exp(e^{im})} \tag{7.2}$$

$$\mathbf{c}^i = \sum_{j=1}^{T_x} \alpha^{i,j} \mathbf{h}^j \tag{7.3}$$

In the APE literature, two recent papers have extended the attention mechanism to contextual APE. [Chatterjee *et al.*, 2017] (the winner of the WMT17 shared task) have proposed a two-encoder system with a separate attention for each encoder. The two attention networks create a context vector for each input, $\mathbf{c}_{src}$ and $\mathbf{c}_{mt}$, and concatenate them using additional, learnable parameters, $\boldsymbol{W}_{ct}$ and $\mathbf{b}_{ct}$, into a merged context vector, $\mathbf{c}_{merge}$ (Equation 7.4).

$$\mathbf{c}^i_{merge} = [\mathbf{c}^i_{src}; \mathbf{c}^i_{mt}] * \boldsymbol{W}_{ct} + \mathbf{b}_{ct} \tag{7.4}$$

[Libovickỳ and Helcl, 2017] have proposed, among others, an attention strategy named the *flat attention*. In this approach, all the attention weights corresponding to the tokens in the two inputs are computed with a joint soft-max:

$$\alpha^{ij}_{(k)} = \frac{exp(e^{ij}_{(k)})}{\sum_{n=1}^{2} \sum_{m=1}^{T_x^{(n)}} exp(e^{im}_{(n)})} \tag{7.5}$$

where $e^{ij}_{(k)}$ is the attention energy of the j-th step of the k-th encoder at the i-th decoding step and $T_x^{(k)}$ is the length of the input sequence of the k-th encoder. Note that because the attention weights are computed jointly over the different encoders, this approach allows observing whether the system assigns more weight to the tokens of the *src* or the *mt* at each decoding step. Once the attention weigths are computed, a single context vector (**c**) is created as:

$$\mathbf{c}^i = \sum_{k=1}^{N} \sum_{j=1}^{T_x^{(k)}} \alpha_{(k)}^{i,j} \mathbf{U}_{c(k)} \mathbf{h}_{(k)}^{j} \tag{7.6}$$

where $\mathbf{h}_{(k)}^{j}$ is the j-th hidden vector from the k-th encoder, $T_x^{(k)}$ is the number of hidden vectors from the k-th encoder, and $\mathbf{U}_{c(k)}$ is the projection matrix for the k-th encoder that projects its hidden vectors to a common-dimensional space. This parameter is also learnable and can further re-weigh the two inputs.

7.3 The proposed model

The main focus of our chapter is on the interpretability of the predictions made by neural APE systems. To this aim, we have assembled a contextual neural model that leverages two encoders and a shared attention mechanism, similarly to the *flat attention* of [Libovický and Helcl, 2017]. To describe it, let us assume that $\mathbf{X}_{src} = \{\mathbf{x}_{src}^1, ..., \mathbf{x}_{src}^N\}$ is the *src* sentence and $\mathbf{X}_{mt} = \{\mathbf{x}_{mt}^1, ..., \mathbf{x}_{mt}^M\}$ is the *mt* sentence, where N and M are their respective numbers of tokens. The two encoders encode the two inputs separately:

$$\begin{aligned} \mathbf{h}_{src}^j &= enc_{src}(\mathbf{x}_{src}^j, \mathbf{h}_{src}^{j-1}) \quad j = 1, ..., N \\ \mathbf{h}_{mt}^j &= enc_{mt}(\mathbf{x}_{mt}^j, \mathbf{h}_{mt}^{j-1}) \quad j = 1, ..., M \end{aligned} \tag{7.7}$$

All the hidden vectors outputs by the two encoders are then concatenated as if they were coming from a single encoder:

$$\mathbf{h}_{join} = \{\mathbf{h}_{src}^1, ..., \mathbf{h}_{src}^N, \mathbf{h}_{mt}^1, ..., \mathbf{h}_{mt}^M\} \tag{7.8}$$

Then, the attention weights and the context vector at each decoding step are computed from the hidden vectors of $\mathbf{h}_{join}$ (Equations 7.9-7.11):

$$e^{ij} = aligment\_model(\mathbf{h}^j_{join}, \mathbf{s}^{i-1}) \tag{7.9}$$

$$\alpha^{ij} = \frac{exp(e^{ij})}{\sum_{m=1}^{N+M} exp(e^{im})} \tag{7.10}$$

$$\mathbf{c}^i = \sum_{j=1}^{N+M} \alpha^{i,j} \mathbf{h}^j_{join} \tag{7.11}$$

where i is the time step on the decoder side, j is the index of the hidden encoded vector. Given that the $\alpha^{i,j}$ weights form a normalized probability distribution over j, this model is "forced" to spread the weight between the *src* and *mt* inputs. Note that our model differs from that proposed by [Libovickỳ and Helcl, 2017] only in that we do not employ the learnable projection matrices, $U_{c(k)}$. This is done to avoid re-weighing the contribution of the two inputs in the context vectors and, ultimately, in the predictions. More details of the proposed model and its hyper-parameters are provided in Section 7.4.3.

7.4 Experiments

7.4.1 Datasets

For training and evaluation we have used the WMT17 APE[1] IT domain English-German dataset. This dataset consists of 11,000 triplets for training, 1,000 for validation and 2,000 for testing. The hyper-parameters have been selected using only the validation set and used unchanged on the test set. We have also trained the model with the 12,000 sentences from the previous year (WMT16), for a total of 23,000 training triplets.

7.4.2 Artificial data

Since the training set provided by the shared task is too small to effectively train neural networks, [Junczys-Dowmunt and Grundkiewicz, 2016] have proposed a method for

[1]http://www.statmt.org/wmt17/ape-task.html

# encoders	2
encoder type	B-LSTM
encoder layers	2
encoder hidden dim	500
# decoders	1
decoder type	LSTM
decoder layers	2
decoder hidden dim	500
word vector dim	300
attention type	*general*
dropout	0.3
beam size	5

Table 7.1: The model and its hyper-parameters.

creating extra, "artificial" training data using round-trip translations. First, a language model of the target language (German here) is learned using a monolingual dataset. Then, only the sentences from the monolingual dataset that have low perplexity are round-trip translated using two off-the-shelf translators (German-English and English-German). The low-perplexity sentences from the monolingual dataset are treated as the *pe*, the German-English translations as the *src*, and the English-German back-translations as the *mt*. Finally, the (*src*, *mt*, *pe*) triplets are filtered to only retain sentences with comparable TER statistics to those of the manually-annotated training data. These artificial data have proved very useful for improving the accuracy of several neural APE systems, and they have therefore been included in the WMT17 APE shared task. In this chapter, we have limited ourselves to using 500K artificial triplets as done in [Varis and Bojar, 2017; Bérard *et al.*, 2017]. To balance artificial and manually-annotated data during training, we have resampled the official 23K triplets 10 times.

7.4.3 Training and hyper-parameters

Hereafter we provide more information about the model's implementation, its hyper-parameters, the pre-processing and the training to facilitate the reproducibility of our results. We have made our code publicly available[2].

To implement the encoder/decoder with separate encoders for the two inputs (*src*, *mt*) and a single attention mechanism, we have modified the open-source OpenNMT code

[2]https://github.com/ijauregiCMCRC/Shared_Attention_for_APE

[Klein *et al.*, 2017].

Table 7.1 lists all hyper-parameters which have all been chosen using only training and validation data. The two encoders have been implemented using a Bidirectional Long Short-Term Memory (B-LSTM) [Hochreiter and Schmidhuber, 1997] while the decoder uses a unidirectional LSTM. Both the encoders and the decoder use two hidden layers. For the attention network, we have used the OpenNMT's *general* option [Luong *et al.*, 2015].

As for the pre-processing, the datasets come already tokenized. Given that German is a morphologically rich language, we have learned the subword units using the BPE algorithm [Sennrich *et al.*, 2015] only over the official training sets from the WMT16 and WMT17 IT-domain APE shared task (23,000 sentences). The number of *merge* operations has been set to 30,000 under the intuition that one or two word splits per sentence could suffice. Three separate vocabularies have been used for the (*src*, *mt* and *pe*) sentences. Each vocabulary contains a maximum of 50,000 most-frequent subword units; the remaining tokens are treated as unknown (<*unk*>).

As mentioned in Section 4.2, we have trained our model with 500K extra triplets as in [Bérard *et al.*, 2017]. We have oversampled the 23K official triplets 10 times, added the extra 500K, and trained the model for 20 epochs. We have used Stochastic Gradien Descent (SGD) with a learning rate of 1 and a learning rate decay of 0.5. The learning rate decays if there are no improvements on the validation set.

In all cases, we have selected the models and hyper-parameters that have obtained the best results on the validation set (1,000 sentences), and reported the results blindly over the test set (2,000 sentences). The performance has been evaluated in two ways following the evaluation carried out in the shared task [Bojar *et al.*, 2017]: first, as common for this task, we have reported the accuracy in terms of Translation Error Rate (TER) [Snover *et al.*, 2006] and BLEU score [Papineni *et al.*, 2002]. Second, we present an empirical analysis of the attention weight matrices for some notable cases.

7.4.4 Results

Table 7.2 compares the accuracy of our model on the test data with two baselines and two state-of-the-art comparable systems. The MT baseline simply consists of the accuracy

Model	TER	BLEU
MT [Bojar *et al.*, 2017]	24.48	62.49
SPE [Bojar *et al.*, 2017]	24.69	62.97
[Varis and Bojar, 2017]	24.03	64.28
[Bérard *et al.*, 2017]	22.81	65.91
train 11K	41.58	43.05
train 23K	30.23	57.14
train 23K + 500K	**22.60**	**66.21**

Table 7.2: Results on the WMT17 IT domain English-German APE test set.

of the *mt* sentences with respect to the *pe* ground truth. The other baseline is given by a statistical PE (SPE) system [Simard *et al.*, 2007] chosen by the WMT17 organizers. Table 7.2 shows that when our model is trained with only the 11K WMT17 official training sentences, it cannot even approach the baselines. Even when the 12K WMT16 sentences are added, its accuracy is still well below that of the baselines. However, when the 500K artificial data are added, it reports a major improvement and it outperforms them both significantly. In addition, we have compared our model with two recent systems that have participated in the shared task and have used our same training settings (500K artificial triplets + 23K manual triplets oversampled 10 times, hence they are directly comparable to our approach), reporting a slightly higher accuracy than both (1.43 TER and 1.93 BLEU p.p. over [Varis and Bojar, 2017] and 0.21 TER and 0.30 BLEU p.p. over [Bérard *et al.*, 2017]). Since their models explicitly predicts edit operations rather than post-edited sentences, we speculate that these two tasks are of comparable intrinsic complexity.

In addition to experimenting with the proposed model (Equation 5.3), we have also tried to add the projection matrices of the flat attention of [Libovickỳ and Helcl, 2017] (Equation 7.6). However, the model with these extra parameters showed evident overfitting, with a lower perplexity on the training set, but unfortunately also a lower BLEU score of 53.59 on the test set. On the other hand, [Chatterjee *et al.*, 2017] and other participants of the WMT 17 APE shared task [3] were able to achieve higher accuracies by using 4 million artificial training triplets. Unfortunately, using such a large dataset sent the computation out of memory on a system with 32 GB of RAM. Nonetheless, our main goal is not to establish the highest possible accuracy, but rather contribute to the interpretability of APE predictions while reproducing approximately the same accuracy

[3] http://www.statmt.org/wmt17/ape-task.html

of current systems trained in a comparable way.

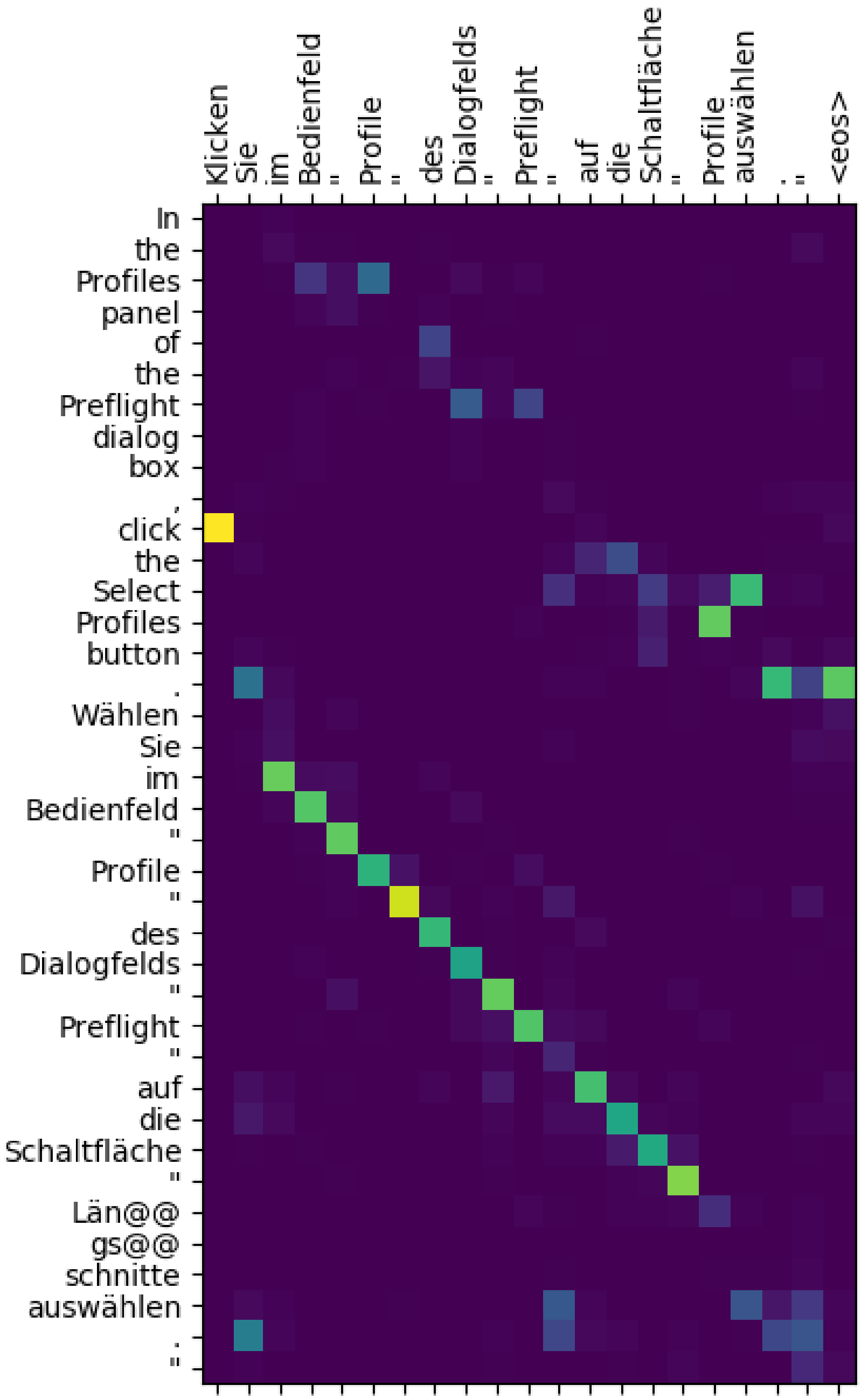


Figure 7.1: An example of perfect correction of an *mt* sentence.

For the analysis of the interpretability of the system, we have plotted the attention weight matrices for a selection of cases from the test set. These plots aim to show how the shared attention mechanism shifts the attention weights between the tokens of the *src* and *mt* inputs at each decoding step. In the matrices, the rows are the concatenation of the *src* and *mt* sentences, while the columns are the predicted *pe* sentence. To avoid cluttering, the ground-truth *pe* sentences are not shown in the plots, but they are commented upon in the discussion. Figure 7.1 shows an example where the *mt* sentence is almost correct. In this example, the attention focuses on passing on the correct part. However, the start (*Wählen*) and end (*Längsschnitte*) of the *mt* sentence are wrong: for these tokens, the model learns to place more weight on the English sentence (*click* and *Select Profiles*). The predicted *pe* is eventually identical to the ground truth.

Conversely, Figure 7.2 shows an example where the *mt* sentence is rather incorrect. In this case, the model learns to focus almost completely on the English sentence, and the prediction is very aligned with it. The predicted *pe* is not identical to the ground truth, but it is significantly more accurate than the *mt*. Figure 7.3 shows a case of a perfect *mt* translation where the model simply learns to pass the sentence on word by word. Eventually, Figure 7.4 shows an example of a largely incorrect *mt* where the model has not been able to properly edit the translation. In this case, the attention matrix is scattered and defocused.

In addition to the visualizations of the attention weights, we have computed an attention statistic over the test set to quantify the proportions of the two inputs. At each decoding time step, we have added up the attention weights corresponding to the *src* input ($\alpha^i_{src} = \sum_{j=1}^{N} \alpha^{ij}$) and those corresponding to the *mt* ($\alpha^i_{mt} = \sum_{j=N+1}^{N+M} \alpha^{ij}$). Note that, obviously, $\alpha^i_{src} + \alpha^i_{mt} = 1$. Then, we have set an arbitrary threshold, $t = 0.6$, and counted step i to the *src* input if $\alpha^i_{src} > t$. If instead $\alpha^i_{src} < 1 - t$, we counted the step to the *mt* input. Eventually, if $1 - t \leq \alpha^i_{src} \leq t$, we counted the step to both inputs. Table 7.3 shows this statistic. Overall, we have recorded 23% decoding steps for the *src*, 45% for the *mt* and 31% for both. It is to be expected that the majority of the decoding steps would focus on the *mt* input if it is of sufficient quality. However, the percentage of focus on the *src* input is significant, confirming its usefulness.

Sentence	**Focus**
src	23%
mt	45%
Both	31%

Table 7.3: Percentage of the decoding steps with marked attention weight on either input (*src*, *mt*) or both.

7.5 Conclusion

In this chapter, we have presented a neural APE system based on two separate encoders that share a single, joined attention mechanism. The shared attention has proved a key feature for inspecting how the selection shifts on either input, (*src* and *mt*), at each decoding step and, in turn, understanding which inputs drive the predictions. In addition to its easy interpretability, our model has reported a competitive accuracy compared to recent, similar systems (i.e., systems trained with the official WMT16 and WMT17 data and 500K extra training triplets). As future work, we plan to continue to explore the interpretability of contemporary neural APE architectures.

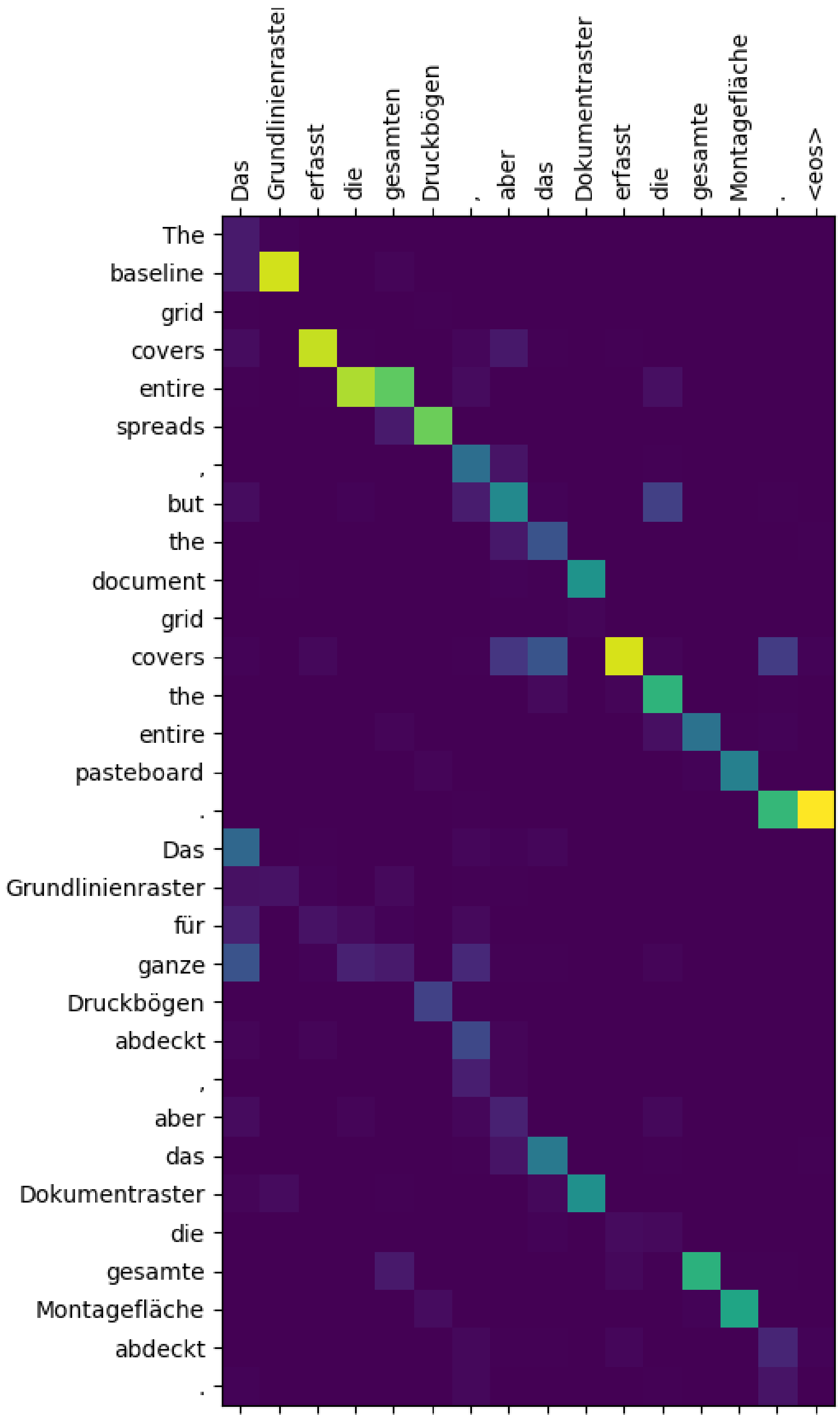


Figure 7.2: Partial improvement of an *mt* sentence.

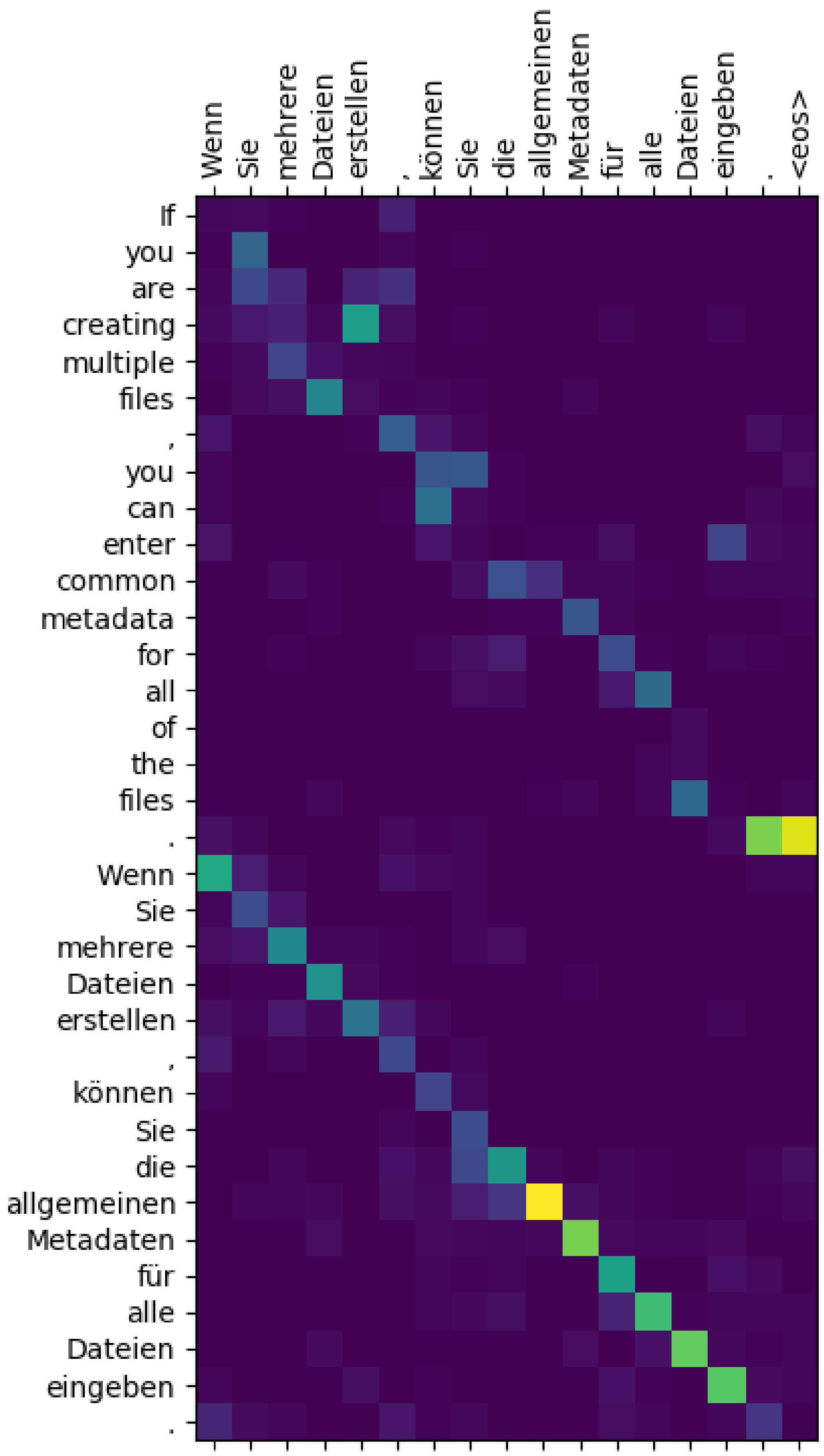


Figure 7.3: Passing on a correct *mt* sentence.

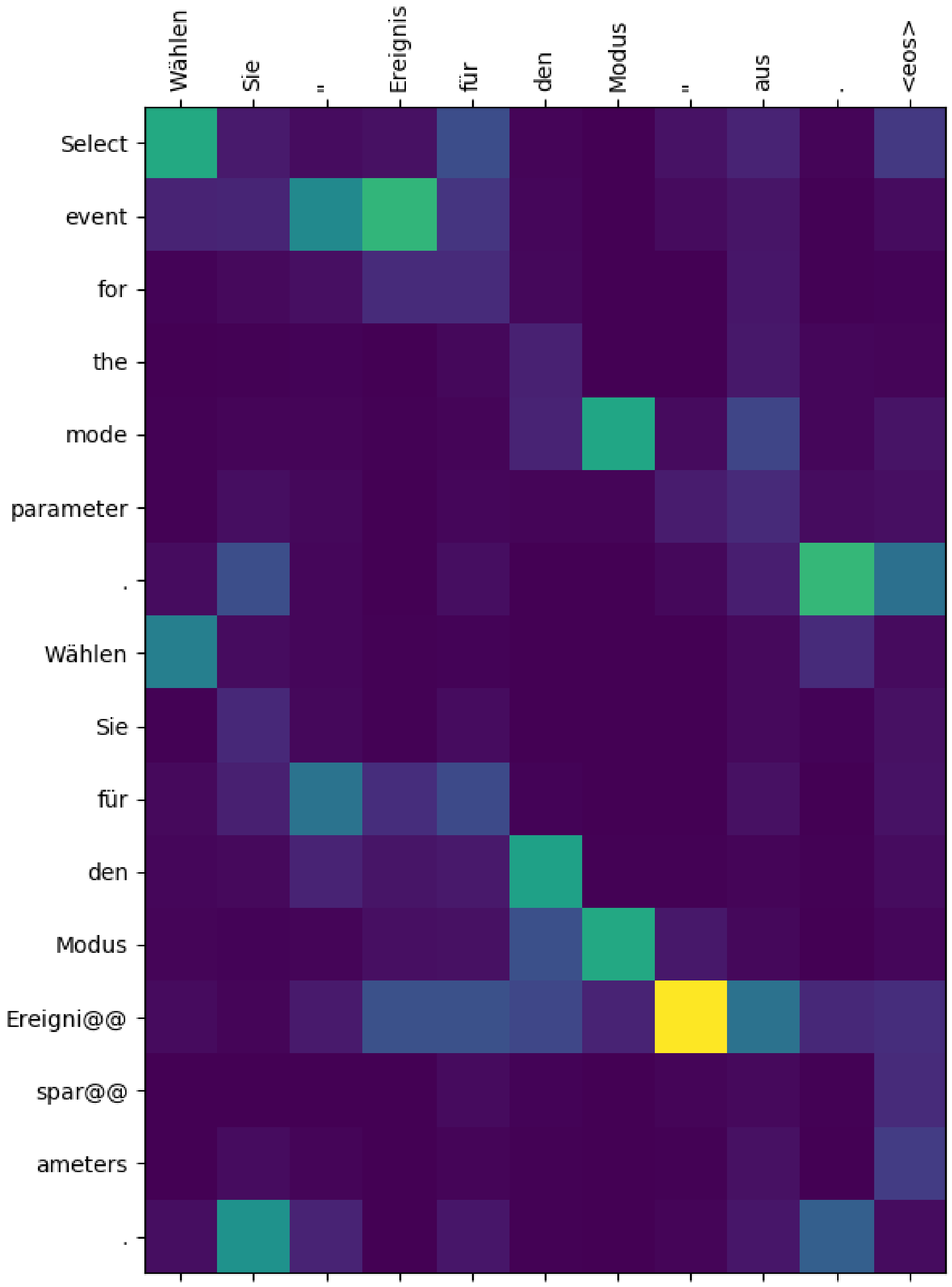


Figure 7.4: A completely incorrect prediction.

Chapter 8

Conclusion

NER and MT are two well-established NLP tasks which have greatly benefited from the recent uptake of state-of-the-art deep learning models. However, through this thesis we have seen that these models are prone to suffer from overfitting when the annotated training datasets are small. This thesis has analyzed this problem in depth and proposed a number of novel approaches to alleviate it.

First, in NER, we have worked with health-domain data. We have used two datasets popular in the research community, one for drug name recognition and the other for clinical concept extraction. Previous work had mainly focused on using traditional sequential classifiers such as HMMs, S-SVMs and CRFs with rich feature engineering. Conversely, we have chosen to use the BiLSTM-CRF deep recurrent neural network given its widely reported accuracy on a variety of other tasks. In order to overcome the scarcity of data, we have pre-trained health-domain specialized word embeddings using a publicly available dataset of ICU patient records, MIMIC-III. Several experiments and ablation studies have shown that the specialized embeddings have been key in outperforming the traditional machine learning models.

Second, in neural machine translation, we have carried out the first systematic performance comparison between traditional SMT models and the more recent NMT models over English-Basque (a low -resource language pair). Our study has shown that SMT models can still outperform current NMT systems when the amount of supervised training data is limited. Nevertheless, our work have also shown that both models still lack strong generalization capability, even in-domain. More work will be needed to improve the MT accuracy, either on the data or the models side.

The following chapters (Chapters 5-7) have focused on improving NMT models for low-resource language pairs and translation domains. We have proposed a novel regularization technique based on regressing word and sentence embeddings. This has allowed us to leverage the semantic and distributional properties of pre-trained embeddings in the objective function. Our experiments have shown how this regularization method can help achieve better translation quality. A qualitative analysis has also shown that the proposed technique contributes to regularizing the decoder's hidden vectors and, in turn, selecting the correct words in the translation.

In chapter 6 we have proposed a novel way of training NMT models for document-level translations that combines a well-established reinforcement-style approach with existing discourse rewards. We have shown that the proposed objective function can improve the lexical cohesion and coherence of the translated documents. Moreover, we have shown that this approach does not come at the expense of, and can even improve, accuracy-based metrics such as the BLEU score.

In chapter 7 we have presented a neural APE model that can learn to edit the errors made by an existing MT model with significantly smaller training datasets. Our model uses a shared attention network across two parallel encoders for the source sentence and the MT output. The proposed approach has achieved a considerable improvement in terms of BLEU score in two IT-domain and health-domain translation tasks. Furthermore, such a shared attention makes the decisions made the neural model more easily interpretable.

Many directions would be interesting for future research. To name one, it would be interesting to explore better rewards for reinforcement-like training in NER and MT models. Particularly, if we could manage to compute the rewards in an unsupervised manner, we would be able to train and tune the models with less effort. It would also be interesting to combine transfer learning (e.g. pre-trained language models) with reinforcement learning, potentially speeding up the convergence of models trained with reinforcement learning objective functions.

Finally, I would like to highlight that all the code developed during this PhD has been made publicly available to facilitate the continuing research in this area. The interested reader can find all the code in the following GitHub repository:

https://github.com/ijauregiCMCRC/

References

[Abacha *et al.*, 2015] Asma Ben Abacha, Md Faisal Mahbub Chowdhury, Aikaterini Karanasiou, Yassine Mrabet, Alberto Lavelli, and Pierre Zweigenbaum. Text mining for pharmacovigilance: Using machine learning for drug name recognition and drug-drug interaction extraction and classification. *Journal of Biomedical Informatics*, 58:122–132, 2015.

[Abadi *et al.*, 2016] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Synposium on Operating Systems Design and Implementation*, volume 16, pages 265–283, 2016.

[Abeillé *et al.*, 1990] Anne Abeillé, Yves Schabes, and Aravind K Joshi. Using lexicalized tags for machine translation. In *Proceedings of the 13th International Conference on Computational Linguistics*, pages 1–6, 1990.

[Agrawal *et al.*, 2018] Ruchit Rajeshkumar Agrawal, Marco Turchi, and Matteo Negri. Contextual handling in neural machine translation: Look behind, ahead and on both sides. In *Proceedings of the 21st Annual Conference of the European Association for Machine Translation*, pages 11–20, 2018.

[Aguilar *et al.*, 2017] Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 2017.

[Aikhenvald, 2007] Alexandra Y Aikhenvald. Typological distinctions in word-formation. *Language typology and syntactic description*, 3:1–65, 2007.

[Antonova and Misyurev, 2011] Alexandra Antonova and Alexey Misyurev. Building a web-based parallel corpus and filtering out machine-translated text. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 136–144, 2011.

[Appelo, 1986] Lisette Appelo. A compositional approach to the translation of temporal expressions in the rosetta system. In *Proceedings of the 11th International Conference on Computational Linguistics*, 1986.

[Arnold and Sadler, 1992] Doug Arnold and Louisa Sadler. Rationalism and the treatment of referential dependencies. In *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation*, 1992.

[Aronson and Lang, 2010] Alan R Aronson and François-Michel Lang. An overview of metamap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229–236, 2010.

[Artetxe and Schwenk, 2019a] Mikel Artetxe and Holger Schwenk. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, page 3197–3203, 2019.

[Artetxe and Schwenk, 2019b] Mikel Artetxe and Holger Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610, 2019.

[Artetxe *et al.*, 2018] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. In *Proceedings of the International Conference on Learning Representations*, 2018.

[Asahara and Matsumoto, 2003] Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 8–15, 2003.

[Ba and Frey, 2013] Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3084–3092, 2013.

[Ba *et al.*, 2016] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Systems*, 2015.

[Bahdanau *et al.*, 2017] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. *Proceedings of the International Conference on Learning Representations*, 2017.

[Banerjee and Lavie, 2005] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, 2005.

[Barbosa *et al.*, 2012] Luciano Barbosa, Vivek Kumar Rangarajan Sridhar, Mahsa Yarmohammadi, and Srinivas Bangalore. Harvesting parallel text in multiple languages with limited supervision. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 201–214, 2012.

[Barrault *et al.*, 2019] Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation*, pages 1–61, 2019.

[Barzilay and Elhadad, 1999] Regina Barzilay and Michael Elhadad. Using lexical chains for text summarization. *Advances in Automatic Text Summarization*, pages 111–121, 1999.

[Barzilay and Lapata, 2008] Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34, 2008.

[Beaven, 1992] John L Beaven. *Lexicalist unification-based machine translation*. PhD thesis, University of Edinburgh, 1992.

[Béchara *et al.*, 2011] Hanna Béchara, Yanjun Ma, and Josef van Genabith. Statistical post-editing for a statistical mt system. In *Machine Translation Summit*, volume 13, pages 308–315, 2011.

[Belinkov *et al.*, 2020] Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. On the linguistic representational power of neural machine translation models. *Computational Linguistics*, 46(1):1–52, 2020.

[Bengio *et al.*, 1994] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[Bengio *et al.*, 2015] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[Bentivogli *et al.*, 2016] Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus phrase-based machine translation quality: a case study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2016.

[Bérard *et al.*, 2017] Olivier Bérard, Alexandre Pietquin, and Laurent Besacier. Lig-cristal system for the wmt17 automatic post-editing task. In *Proceedings of the Second Conference on Machine Translation*, pages 623–629, 2017.

[Bergstra and Bengio, 2012] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[Bergstra *et al.*, 2010] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: A cpu and gpu math compiler in python. In *Proceedings of the 9th Python in Science Conference*, volume 1, 2010.

[Bikel *et al.*, 1997] Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the Conference on Applied Natural Language Processing*, 1997.

[Boag *et al.*, 2015] William Boag, Kevin Wacome, Tristan Naumann, and Anna Rumshisky. Cliner: A lightweight tool for clinical named entity recognition. In *Proceedings of the American Medical Informatics Association Joint Summits on Clinical Research Informatics (poster)*, 2015.

[Bojanowski *et al.*, 2017] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[Bojar *et al.*, 2013] Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, 2013.

[Bojar *et al.*, 2016] Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*, volume 2, pages 131–198, 2016.

[Bojar *et al.*, 2017] Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. Findings of the 2017 conference on machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, 2017.

[Bojar *et al.*, 2018] Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, 2018.

[Borthwick *et al.*, 1998] Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. Nyu: Description of the mene named entity system as used in muc-7. In *Proceedings of the 7th Message Understanding Conference*, 1998.

[Bosselut *et al.*, 2018] Antoine Bosselut, Asli Celikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. Discourse-aware neural rewards for coherent text generation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2018.

[Bowman *et al.*, 2015] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 632–642, 2015.

[Brin, 1998] Sergey Brin. Extracting patterns and relations from the world wide web. In *Proceedings of the International Workshop on The World Wide Web and Databases*, pages 172–183, 1998.

[Britton and Gülgöz, 1991] Bruce K Britton and Sami Gülgöz. Using kintsch's computational model to improve instructional text: Effects of repairing inference calls on recall and cognitive structures. *Journal of educational Psychology*, 83(3):329, 1991.

[Brown *et al.*, 1990] Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Frederik Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

[Brown *et al.*, 1992] Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480, 1992.

[Callison-Burch *et al.*, 2006] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy, April 2006. Association for Computational Linguistics.

[Carbonell *et al.*, 1978] Jaime G Carbonell, Richard E Cullinford, and Anatole V Gershman. Knowledge-based machine translation. Technical report, Yale University, Department of Computer Science, 1978.

[Castilho *et al.*, 2017] Sheila Castilho, Joss Moorkens, Federico Gaspari, Rico Sennrich, Vilelmini Sosoni, Panayota Georgakopoulou, Pintu Lohar, Andy Way, Antonio Valerio Miceli Barone, and Maria Gialama. A comparative quality evaluation of pbsmt and nmt using professional translators. 2017.

[Cer *et al.*, 2017] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. In *Proceedings of the SemEval Workshop*, 2017.

[Cer *et al.*, 2018] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[Chalapathy *et al.*, 2016a] Raghavendra Chalapathy, Ehsan Zare Borzeshi, and Massimo Piccardi. Bidirectional lstm-crf for clinical concept extraction. In *Proceedings of the Clinical Natural Language Processing Workshop*, 2016.

[Chalapathy *et al.*, 2016b] Raghavendra Chalapathy, Ehsan Zare Borzeshi, and Massimo Piccardi. An investigation of recurrent neural architectures for drug name recognition. In *Proceedings of the 7th International Workshop on Health Text Mining and Information Analysis*, 2016.

[Changpinyo *et al.*, 2018] Soravit Changpinyo, Hexiang Hu, and Fei Sha. Multi-task learning for sequence tagging: An empirical study. In *Proceedings of the 27th International Conference on Computational Linguistics*, 2018.

[Chatterjee *et al.*, 2017] Rajen Chatterjee, M Amin Farajian, Matteo Negri, Marco Turchi, Ankit Srivastava, and Santanu Pal. Multi-source neural automatic post-editing: Fbk's participation in the wmt 2017 ape shared task. In *Proceedings of the Second Conference on Machine Translation*, pages 630–638, 2017.

[Chen *et al.*, 2004] Jisong Chen, Rowena Chau, and Chung-Hsing Yeh. Discovering parallel text from the world wide web. In *Proceedings of the 2nd Workshop on Australasian Information Security, Data Mining and Web Intelligence, and Software Internationalisation*, pages 157–161, 2004.

[Chinchor *et al.*, 1998] Nancy Chinchor, Patty Robinson, and Erica Brown. Hub-4 named entity task definition version 4.8. *Available by ftp from www.nist.gov/speech/hub4_98*, 1998.

[Chinea-Rios *et al.*, 2017] Mara Chinea-Rios, Alvaro Peris, and Francisco Casacuberta. Adapting neural machine translation with parallel synthetic data. In *Proceedings of the Second Conference on Machine Translation*, pages 138–147, 2017.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014.

[Chomsky, 1956] Noam Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124, 1956.

[Chousa *et al.*, 2018] Katsuki Chousa, Katsuhito Sudoh, and Satoshi Nakamura. Training neural machine translation using word embedding-based loss. *arXiv:1807.11219 [cs]*, 2018.

[Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of the Deep Learning and Representation Learning Workshop*, 2014.

[Clark *et al.*, 2018] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

[Cocos *et al.*, 2017] Anne Cocos, Alexander G Fiks, and Aaron J Masino. Deep learning for pharmacovigilance: recurrent neural network architectures for labeling adverse drug reactions in twitter posts. *Journal of the American Medical Informatics Association*, 24(4):813–821, 2017.

[Collins and Singer, 1999] Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

[Collobert and Weston, 2008] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, 2008.

[Collobert *et al.*, 2002] Ronan Collobert, Samy Bengio, and Johnny Mariéthoz. Torch: a modular machine learning software library. Technical report, Idiap, 2002.

[Conneau *et al.*, 2017] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017.

[Coulet *et al.*, 2011] Adrien Coulet, Yael Garten, Michel Dumontier, Russ B Altman, Mark A Musen, and Nigam H Shah. Integration and publication of heterogeneous text-mined relationships on the semantic web. In *Journal of biomedical semantics*, volume 2, page S10. Springer, 2011.

[Crichton *et al.*, 2017] Gamal Crichton, Sampo Pyysalo, Billy Chiu, and Anna Korhonen. A neural network multi-task learning approach to biomedical named entity recognition. *BMC Bioinformatics*, 18(1):368, 2017.

[Currey *et al.*, 2017] Anna Currey, Antonio Valerio Miceli-Barone, and Kenneth Heafield. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156, 2017.

[Daumé *et al.*, 2009] Hal Daumé, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine Learning*, 75(3):297–325, 2009.

[de Bruijn *et al.*, 2011] Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Xiaodan Zhu. Machine-learned solutions for three stages of clinical information extraction: the state of the art at i2b2 2010. *Journal of the American Medical Informatics Association*, 18(5):557–562, 2011.

[de Ilarraza *et al.*, 2008] A Diaz de Ilarraza, Gorka Labaka, and Kepa Sarasola. Statistical postediting: A valuable method in domain adaptation of rbmt systems for less-resourced languages. *Mixing Approaches to Machine Translation*, pages 35–40, 2008.

[De Matos *et al.*, 2010] Paula De Matos, Rafael Alcántara, Adriano Dekker, Marcus Ennis, Janna Hastings, Kenneth Haug, Inmaculada Spiteri, Steve Turner, and Christoph Steinbeck. Chemical entities of biological interest: an update. *Nucleic acids research*, 38(suppl_1):D249–D254, 2010.

[Del Gaudio *et al.*, 2016] Rosa Del Gaudio, Gorka Labaka, Eneko Agirre, Petya Osenova, Kiril Ivanov Simov, Martin Popel, Dieke Oele, Gertjan van Noord, Luís Gomes, João António Rodrigues, et al. Smt and hybrid systems of the qtleap project in the wmt16 it-task. In *Proceedings of the First Conference on Machine Translation*, pages 435–441, 2016.

[Deng *et al.*, 2013] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8599–8603, 2013.

[Denkowski and Neubig, 2017] Michael Denkowski and Graham Neubig. Stronger baselines for trustable results in neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 18–27, 2017.

[Dernoncourt *et al.*, 2017] Franck Dernoncourt, Ji Young Lee, Ozlem Uzuner, and Peter Szolovits. De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 24(3):596–606, 2017.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2019.

[Ding *et al.*, 2017] Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, 2017.

[Doddington *et al.*, 2004] George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Proceedings of the 4th Language Resources and Evaluation Conference*, volume 2, page 1, 2004.

[Doddington, 2002] George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the 2nd International Conference on Human Language Technology Research*, pages 138–145, 2002.

[Doi and Muraki, 1992] Shinichi Doi and Kazunori Muraki. Translation ambiguity resolution based on text corpora of source and target languages. In *Proceedings of the 15th International Conference on Computational Linguistics*, 1992.

[Domhan and Hieber, 2017] Tobias Domhan and Felix Hieber. Using target-side monolingual data for neural machine translation through multi-task learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1500–1505, 2017.

[Dorr, 1991] Bonnie Jean Dorr. Principle-based parsing for machine translation. In *Principle-Based Parsing*, pages 153–183. Springer, 1991.

[Duong *et al.*, 2015] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 845–850, 2015.

[Edunov *et al.*, 2018a] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

[Edunov *et al.*, 2018b] Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2018.

[Edunov *et al.*, 2019] Sergey Edunov, Alexei Baevski, and Michael Auli. Pre-trained language model representations for language generation. *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2019.

[Elbayad *et al.*, 2018] Maha Elbayad, Laurent Besacier, and Jakob Verbeek. Token-level and sequence-level loss smoothing for RNN language models. In *Procedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2094–2103, 2018.

[ElJundi *et al.*, 2019] Obeida ElJundi, Wissam Antoun, Nour El Droubi, Hazem Hajj, Wassim El-Hajj, and Khaled Shaban. Hulmona: The universal language model in arabic. In *Proceedings of the 4th Arabic Natural Language Processing Workshop*, pages 68–77, 2019.

[Elman, 1990] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[Espla-Gomis and Forcada, 2010] Miquel Espla-Gomis and Mikel Forcada. Combining content-based and url-based heuristics to harvest aligned bitexts from multilingual sites with bitextor. *The Prague Bulletin of Mathematical Linguistics*, 93:77–86, 2010.

[Espla-Gomis *et al.*, 2014] Miquel Espla-Gomis, Filip Klubicka, Nikola Ljubesic, Sergio Ortiz-Rojas, Vassilis Papavassiliou, and Prokopis Prokopidis. Comparing two acquisition systems for automatically building an english-croatian parallel corpus from multilingual websites. In *Proceedings of the 9th Language Resources and Evaluation Conference*, pages 1252–1258, 2014.

[Fadaee and Monz, 2018] Marzieh Fadaee and Christof Monz. Back-translation sampling by targeting difficult words in neural machine translation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

[Fadaee *et al.*, 2017] Marzieh Fadaee, Arianna Bisazza, and Christof Monz. Data augmentation for low-resource neural machine translation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[Fellbaum, 2012] Christiane Fellbaum. Wordnet. *The encyclopedia of applied linguistics*, 2012.

[Foltz *et al.*, 1998] Peter W Foltz, Walter Kintsch, and Thomas K Landauer. The measurement of textual coherence with latent semantic analysis. *Discourse processes*, 25(2-3):285–307, 1998.

[Fu and Ananiadou, 2014] Xiao Fu and Sophia Ananiadou. Improving the extraction of clinical concepts from clinical records. In *Proceedings of the 4th Workshop on Building and Evaluating Resources for Health and Biomedical Text Processing*, 2014.

[Gal and Ghahramani, 2016] Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1019–1027, 2016.

[Gao *et al.*, 2019] Jianfeng Gao, Michel Galley, Lihong Li, et al. Neural approaches to conversational ai. *Foundations and Trends® in Information Retrieval*, 13(2-3):127–298, 2019.

[Girshick, 2015] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[Glorot and Bengio, 2010] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[Gong *et al.*, 2015] Zhengxian Gong, Min Zhang, and Guodong Zhou. Document-level machine translation evaluation with gist consistency and text cohesion. In *Proceedings of the Second Workshop on Discourse in Machine Translation*, pages 33–40, 2015.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[Graves, 2012] Alex Graves. Supervised sequence labelling with recurrent neural networks. *Studies in Computational Intelligence*, 2012.

[Gridach, 2017] Mourad Gridach. Character-level neural network for biomedical named entity recognition. *Journal of biomedical informatics*, 70:85–91, 2017.

[Grishman and Sundheim, 1996] Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics*, 1996.

[Grosz *et al.*, 1995] Barbara J Grosz, Scott Weinstein, and Aravind K Joshi. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics*, 21(2):203–225, 1995.

[Gu *et al.*, 2018] Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, and Victor OK Li. Meta-learning for low-resource neural machine translation. *Proceedings of the Conference on Empirical Methods in Natural Languae Processing*, 2018.

[Gutmann and Hyvärinen, 2012] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.

[Hajlaoui and Popescu-Belis, 2013] Najeh Hajlaoui and Andrei Popescu-Belis. Assessing the accuracy of discourse connective translations: Validation of an automatic metric. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics*, pages 236–247, 2013.

[Halliday and Hasan, 2014] Michael Alexander Kirkwood Halliday and Ruqaiya Hasan. *Cohesion in english*. Routledge, 2014.

[Hardmeier and Federico, 2010] Christian Hardmeier and Marcello Federico. Modelling pronominal anaphora in statistical machine translation. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 283–289, 2010.

[Harkema *et al.*, 2009] Henk Harkema, John N Dowling, Tyler Thornblade, and Wendy W Chapman. Context: an algorithm for determining negation, experiencer, and temporal status from clinical reports. *Journal of biomedical informatics*, 42(5):839–851, 2009.

[Harris, 1988] Brian Harris. Bi-text, a new concept in translation theory. *Language Monthly*, 54(March):8–10, 1988.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[Heafield, 2011] Kenneth Heafield. Kenlm: Faster and smaller language model queries. In *Proceedings of the 6th Workshop on Statistical Machine Translation*, pages 187–197, 2011.

[Hearst, 1997] Marti A Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33–64, 1997.

[Herrero-Zazo *et al.*, 2013] María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, and Thierry Declerck. The ddi corpus: An annotated corpus with pharmacological substances and drug–drug interactions. *Journal of biomedical informatics*, 46(5):914–920, 2013.

[Hettne *et al.*, 2009] Kristina M Hettne, Rob H Stierum, Martijn J Schuemie, Peter JM Hendriksen, Bob JA Schijvenaars, Erik M van Mulligen, Jos Kleinjans, and Jan A Kors. A dictionary to identify small molecules and drugs in free text. *Bioinformatics*, 25(22):2983–2991, 2009.

[Hinton *et al.*, 2012] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.

[Hoang *et al.*, 2018] Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, 2018.

[Hobbs, 1979] Jerry R Hobbs. Coherence and coreference. *Cognitive science*, 3(1):67–90, 1979.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Hopfield, 1982] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. In *Proceedings of the National Academy of Sciences*, volume 79, pages 2554–2558, 1982.

[Hu and Liu, 2004] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 168–177, 2004.

[Huang *et al.*, 2015] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[Huang, 2008] Anna Huang. Similarity measures for text document clustering. In *Proceedings of the 6th New Zealand Computer Science Research Student Conference*, volume 4, pages 9–56, 2008.

[Isozaki *et al.*, 2010] Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of Empirical Methods in Natural Language Processing*, 2010.

[Iyyer *et al.*, 2015] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, volume 1, pages 1681–1691, 2015.

[Jagannatha and Yu, 2016] Abhyuday N Jagannatha and Hong Yu. Structured prediction models for rnn based sequence labeling in clinical text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, page 856, 2016.

[Jauregi Unanue *et al.*, 2017] Inigo Jauregi Unanue, Ehsan Zare Borzeshi, and Massimo Piccardi. Recurrent neural networks with specialized word embeddings for health-domain named-entity recognition. *Journal of biomedical informatics*, 76:102–109, 2017.

[Jauregi Unanue *et al.*, 2018a] Inigo Jauregi Unanue, Lierni Garmendia Arratibel, Ehsan Zare Borzeshi, and Massimo Piccardi. English-basque statistical and neural machine translation. In *Proceedings of the 11th Language Resources and Evaluation Conference*, 2018.

[Jauregi Unanue *et al.*, 2018b] Inigo Jauregi Unanue, Ehsan Zare Borzeshi, and Massimo Piccardi. A shared attention mechanism for interpretation of neural automatic post-editing systems. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, 2018.

[Jauregi Unanue *et al.*, 2019] Inigo Jauregi Unanue, Ehsan Zare Borzeshi, Nazanin Esmaili, and Massimo Piccardi. Rewe: Regressing word embeddings for regularization of neural machine translation systems. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2019.

[Jean *et al.*, 2017] Sebastien Jean, Stanislas Lauly, Orhan Firat, and Kyunghyun Cho. Does neural machine translation benefit from larger context? *arXiv preprint arXiv:1704.05135*, 2017.

[Johnson *et al.*, 2016a] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[Johnson *et al.*, 2016b] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's multilingual neural machine translation system: enabling zero-shot translation. *arXiv preprint arXiv:1611.04558*, 2016.

[Jordan, 1990] Michael Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. *IEEE Press*, pages 112–127, 1990.

[Junczys-Dowmunt and Grundkiewicz, 2016] Marcin Junczys-Dowmunt and Roman Grundkiewicz. Log-linear combinations of monolingual and bilingual neural machine translation models for automatic post-editing. In *Proceedings of the First Conference on Machine Translation*, pages 751–558, 2016.

[Jurafsky and Martin, 2019] D Jurafsky and JH Martin. Speech and language processing (3rd (draft) ed.), 2019.

[Kalajahi *et al.*, 2012] Seyed Ali Rezvani Kalajahi, Ain Nadzimah Abdullah, Jayakaran Mukundan, and Dan J Tannacito. Discourse connectors: An overview of the history, definition and classification of the term. *World Applied Sciences Journal*, 19(11):1659–1673, 2012.

[Khayrallah and Koehn, 2018] Huda Khayrallah and Philipp Koehn. On the impact of various types of noise on neural machine translation. *arXiv preprint arXiv:1805.12282*, 2018.

[Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[Kiros *et al.*, 2015] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3294–3302, 2015.

[Klein *et al.*, 2017] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*, 2017.

[Klinger *et al.*, 2008] Roman Klinger, Corinna Kolářik, Juliane Fluck, Martin Hofmann-Apitius, and Christoph M Friedrich. Detection of iupac and iupac-like chemical names. *Bioinformatics*, 2008.

[Kocmi and Bojar, 2018] Tom Kocmi and Ondřej Bojar. Trivial transfer learning for low-resource neural machine translation. In *Proceedings of the Third Conference on Machine Translation*, 2018.

[Koehn and Knowles, 2017] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. In *1st Workshop on Neural Machine Translation*, 2017.

[Koehn and Monz, 2006] Philipp Koehn and Christof Monz. Manual and automatic evaluation of machine translation between european languages. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 102–121, 2006.

[Koehn *et al.*, 2003] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, 2003.

[Koehn *et al.*, 2007] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, 2007.

[Kuang *et al.*, 2018] Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics*, 2018.

[Kudo, 2018] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics)*, pages 66–75, 2018.

[Kumar and Tsvetkov, 2018] Sachin Kumar and Yulia Tsvetkov. Von Mises-Fisher loss for training sequence to sequence models with continuous outputs. *Proceedings of the International Conference on Learning Representations*, 2018.

[Lafferty *et al.*, 2001] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *not known*, 2001.

[Lample *et al.*, 2016] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2016.

[Lample *et al.*, 2018] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

[Lan *et al.*, 2020] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *Proceedings of the International Conference on Learning Representations*, 2020.

[Lang *et al.*, 1990] Kevin J Lang, Alex H Waibel, and Geoffrey E Hinton. A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43, 1990.

[Lawrence *et al.*, 1997] Steve Lawrence, C Lee Giles, and Ah Chung Tsoi. Lessons in neural network training: Overfitting may be harder than expected. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 540–545, 1997.

[Le and Mikolov, 2014] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the International Conference on Machine Learning*, pages 1188–1196, 2014.

[Leaman and Gonzalez, 2008] Robert Leaman and Graciela Gonzalez. Banner: an executable survey of advances in biomedical named entity recognition. In *Biocomputing 2008*, pages 652–663. World Scientific, 2008.

[Leblond *et al.*, 2018] Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. SEARNN: Training RNNs with global-local losses. *Proceedings of the International Conference on Learning Representations*, 2018.

[Lebret and Collobert, 2013] Rémi Lebret and Ronan Collobert. Word emdeddings through hellinger pca. In *Proceedings of the European Chapter of the Association for Computational Linguistics*, 2013.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015.

[Lee *et al.*, 2016] Ji Young Lee, Franck Dernoncourt, Ozlem Uzuner, and Peter Szolovits. Feature-augmented neural networks for patient note de-identification. In *Proceedings of the Clinical Natural Language Processing Workshop*, 2016.

[Li and Jurafsky, 2017] Jiwei Li and Dan Jurafsky. Neural net models for open-domain discourse coherence. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017.

[Li and Roth, 2002] Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational linguistics*, pages 1–7, 2002.

[Li and Yang, 2018] Yang Li and Tao Yang. Word embedding for understanding natural language: a survey. In *Guide to Big Data Applications*, pages 83–104. Springer, 2018.

[Libovickỳ and Helcl, 2017] Jindřich Libovickỳ and Jindřich Helcl. Attention strategies for multi-source sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[Libovický *et al.*, 2016] Jindřich Libovický, Jindřich Helcl, Marek Tlustý, Pavel Pecina, and Ondřej Bojar. Cuni system for wmt16 automatic post-editing and multimodal translation tasks. In *Proceedings of the First Conference on Machine Translation*, 2016.

[Liddy, 2001] Elizabeth D Liddy. Natural language processing. 2001.

[Lin and Hovy, 2000] Chin-Yew Lin and Eduard Hovy. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 495–501, 2000.

[Lison *et al.*, 2018] Pierre Lison, Jörg Tiedemann, Milen Kouylekov, et al. Open subtitles 2018: Statistical rescoring of sentence alignments in large, noisy parallel corpora. In *Proceedings of the 11th Language Resources and Evaluation Conference*, 2018.

[Liu *et al.*, 2015] Shengyu Liu, Buzhou Tang, Qingcai Chen, Xiaolong Wang, and Xiaoming Fan. Feature engineering for drug name recognition in biomedical texts: Feature conjunction and feature selection. *Computational and mathematical methods in medicine*, 2015.

[Liu *et al.*, 2018] Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. Empower sequence labeling with task-aware neural language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5253–5260, 2018.

[Liu *et al.*, 2019] Nelson F Liu, Roy Schwartz, and Noah A Smith. Inoculation by fine-tuning: A method for analyzing challenge datasets. *arXiv preprint arXiv:1904.02668*, 2019.

[Luong *et al.*, 2015] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2015.

[Ma and Hovy, 2016] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

[Ma and Liberman, 1999] Xiaoyi Ma and Mark Liberman. Bits: A method for bilingual text search over the web. In *Proceedings of the 7th Machine Translation Summit*, pages 538–542, 1999.

[Ma *et al.*, 2018] Shuming Ma, Xu Sun, Yizhong Wang, and Junyang Lin. Bag-of-words as target for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 332–338, 2018.

[Maaten and Hinton, 2008] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[Macé and Servan, 2019] Valentin Macé and Christophe Servan. Using whole document context in neural machine translation. In *Proceedings of the International Workshop on Spoken Language Translation*, 2019.

[Màrquez and Rodríguez, 1998] Lluís Màrquez and Horacio Rodríguez. Part-of-speech tagging using decision trees. In *Proceedings of the European Conference on Machine Learning*, pages 25–36, 1998.

[Maruf and Haffari, 2018] Sameen Maruf and Gholamreza Haffari. Document context neural machine translation with memory networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[Maruf *et al.*, 2019a] Sameen Maruf, André FT Martins, and Gholamreza Haffari. Selective attention for context-aware neural machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2019.

[Maruf *et al.*, 2019b] Sameen Maruf, Fahimeh Saleh, and Gholamreza Haffari. A survey on document-level machine translation: Methods and evaluation. *arXiv preprint arXiv:1912.08494*, 2019.

[Mayor *et al.*, 2011] Aingeru Mayor, Inaki Alegria, Arantza Diaz De Ilarraza, Gorka Labaka, Mikel Lersundi, and Kepa Sarasola. Matxin, an open-source rule-based machine translation system for basque. *Machine translation*, 25(1):53, 2011.

[McCallum and Li, 2003] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 188–191, 2003.

[McNamara *et al.*, 1996] Danielle S McNamara, Eileen Kintsch, Nancy Butler Songer, and Walter Kintsch. Are good texts always better? interactions of text coherence, background knowledge, and levels of understanding in learning from text. *Cognition and instruction*, 14(1):1–43, 1996.

[Miculicich and Popescu-Belis, 2017] Lesly Miculicich and Andrei Popescu-Belis. Validation of an automatic metric for the accuracy of pronoun translation (apt). In *Proceedings of the 3rd Workshop on Discourse in Machine Translation*, pages 17–25, 2017.

[Miculicich *et al.*, 2018] Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

[Mihalcea and Strapparava, 2009] Rada Mihalcea and Carlo Strapparava. The lie detector: Explorations in the automatic recognition of deceptive language. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*, pages 309–312, 2009.

[Mikheev, 2003] Andrei Mikheev. Text segmentation. In *The Oxford Handbook of Computational Linguistics 2nd edition*, 2003.

[Mikolov *et al.*, 2013a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[Mikolov *et al.*, 2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[Miller *et al.*, 2004] Scott Miller, Jethran Guinness, and Alex Zamanian. Name tagging with word clusters and discriminative training. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 337–342, 2004.

[Miller, 1998] George A Miller. *WordNet: An electronic lexical database*. MIT press, 1998.

[Mintz *et al.*, 2009] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*, pages 1003–1011, 2009.

[Mnih and Teh, 2012] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[Mori *et al.*, 1999] Shunji Mori, Hirobumi Nishida, and Hiromitsu Yamada. *Optical character recognition*. John Wiley & Sons, Inc., 1999.

[Morin and Bengio, 2005] Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *International Conference on Artificial Intelligence and Statistics*, volume 5, pages 246–252, 2005.

[Morris and Hirst, 1991] Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):21–48, 1991.

[Morwal *et al.*, 2012] Sudha Morwal, Nusrat Jahan, and Deepti Chopra. Named entity recognition using hidden markov model (hmm). *International Journal on Natural Language Computing*, 1(4):15–23, 2012.

[Nadeau and Sekine, 2007] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[Nikfarjam *et al.*, 2015] Azadeh Nikfarjam, Abeed Sarker, Karen O'Connor, Rachel Ginn, and Graciela Gonzalez. Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features. *Journal of the American Medical Informatics Association*, 22(3):671–681, 2015.

[Nomiyama, 1992] Hiroshi Nomiyama. Machine translation by case generalization. In *Proceedings of the 15th International Conference on Computational Linguistics*, 1992.

[Och and Ney, 2003] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51, 2003.

[Pal *et al.*, 2017] Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, Qun Liu, and Josef van Genabith. Neural automatic post-editing using prior alignment and reranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 349–355, 2017.

[Pang and Lee, 2005] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 115–124, 2005.

[Pang *et al.*, 2008] Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[Papavassiliou *et al.*, 2013] Vassilis Papavassiliou, Prokopis Prokopidis, and Gregor Thurmair. A modular open-source focused crawler for mining monolingual and bilingual corpora from the web. In *Proceedings of the 6th Workshop on Building and Using Comparable Corpora*, pages 43–51, 2013.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[Pascanu *et al.*, 2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the International Conference on Machine Learning*, pages 1310–1318, 2013.

[Passban, 2017] Peyman Passban. *Machine translation of morphologically rich languages using deep neural networks*. PhD thesis, Dublin City University, 2017.

[Paulus *et al.*, 2018] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *Proceedings of the International Conference on Learning Representations*, 2018.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[Peters *et al.*, 2017] Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[Peters *et al.*, 2018] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2018.

[Poncelas *et al.*, 2018] Alberto Poncelas, Dimitar Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. Investigating backtranslation in neural machine translation. *arXiv preprint arXiv:1804.06189*, 2018.

[Pradhan *et al.*, 2013] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. Towards robust linguistic analysis using ontonotes. In *Proceedings of the 17th Conference on Computational Natural Language Learning*, pages 143–152, 2013.

[Prechelt, 1998] Lutz Prechelt. Early stopping-but when?. *Neural Networks: Tricks of the trade*, pages 55–69, 1998.

[Radford *et al.*, 2018] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[Ramshaw and Marcus, 1999] Lance A Ramshaw and Mitchell P Marcus. Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Natural Language Processing Using Very Large Corpora*, pages 157–176, 1999.

[Ramsundar *et al.*, 2015] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. Massively multitask networks for drug discovery. *arXiv preprint arXiv:1502.02072*, 2015.

[Ranzato *et al.*, 2015] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *Proceedings of the International Conference on Learning Representations*, 2015.

[Rau, 1991] Lisa F Rau. Extracting company names from text. In *Proceedings of the 7th IEEE Conference on Artificial Intelligence Application*, volume 1, pages 29–32, 1991.

[Reimers and Gurevych, 2019] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2019.

[Rios *et al.*, 2017] Annette Rios, Laura Mascarell, and Rico Sennrich. Improving word sense disambiguation in neural machine translation with sense embeddings. In *Proceedings of the 2nd Conference on Machine Translation*, pages 11–19, 2017.

[Rocktäschel *et al.*, 2012] Tim Rocktäschel, Michael Weidlich, and Ulf Leser. Chemspot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640, 2012.

[Rocktäschel *et al.*, 2013] Tim Rocktäschel, Torsten Huber, Michael Weidlich, and Ulf Leser. Wbi-ner: The impact of domain-specific features on the performance of identifying and classifying mentions of drugs. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, volume 2, pages 356–363, 2013.

[Ross *et al.*, 2011] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Conference on Artificial Intelligence and Statistic*, pages 627–635, 2011.

[Ruder, 2017] Sebastian Ruder. An overview of multi task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[Rumelhart *et al.*, 1986] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

[San Vicente *et al.*, 2012] Inaki San Vicente, Iker Manterola, et al. Paco2: A fully automated tool for gathering parallel corpora from the web. In *Proceedings of the 8th Language Resources and Evaluation Conference*, pages 1–6, 2012.

[Sang and De Meulder, 2003] Erik F Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Natural Language Learning*, 2003.

[Sang and Veenstra, 1999] Erik F Sang and Jorn Veenstra. Representing text chunks. *arXiv preprint cs/9907006*, 1999.

[Santos *et al.*, 2006] Diana Santos, Nuno Seco, Nuno Cardoso, and Rui Vilela. Harem: An advanced ner evaluation contest for portuguese. In *Proceedings of the 5th Language Resources and Evaluation Conference*, 2006.

[Savova *et al.*, 2010] Guergana K Savova, James J Masanz, Philip V Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C Kipper-Schuler, and Christopher G Chute. Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

[Scherrer *et al.*, 2019] Yves Scherrer, Jörg Tiedemann, and Sharid Loáiciga. Analysing concatenation approaches to document-level nmt in two different domains. In *Proceedings of the 4th Workshop on Discourse in Machine Translation*, pages 51–61, 2019.

[Schuster and Paliwal, 1997] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[Segura-Bedmar *et al.*, 2015] Isabel Segura-Bedmar, Víctor Suárez-Paniagua, and Paloma Martínez. Exploring word embedding for drug name recognition. In *Proceedings of the 6th International Workshop on Health Text Mining and Information Analysis*, pages 64–72, 2015.

[Sennrich *et al.*, 2015] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2015.

[Sennrich *et al.*, 2016] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

[Settles, 2005] Burr Settles. Abner: an open source tool for automatically tagging genes, proteins and other entity names in text. *Bioinformatics*, 21(14):3191–3192, 2005.

[Shen *et al.*, 2016] Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

[Shu *et al.*, 2017] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36, 2017.

[Simard *et al.*, 2007] Michel Simard, Cyril Goutte, and Pierre Isabelle. Statistical phrase-based post-editing. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 505–515, 2007.

[Snover *et al.*, 2006] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the Association for Machine Translation in the Americas*, 2006.

[Socher *et al.*, 2011] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the International Conference on Machine Learning*, pages 129–136, 2011.

[Søgaard and Goldberg, 2016] Anders Søgaard and Yoav Goldberg. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 231–235, 2016.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Stefanescu *et al.*, 2014] Dan Stefanescu, Rajendra Banjade, and Vasile Rus. Latent semantic analysis models on wikipedia and tasa. In *Proceedings of the 9th Language Resources and Evaluation Conference*, 2014.

[Stroppa *et al.*, 2006] Nicolas Stroppa, Declan Groves, Andy Way, and Kepa Sarasola. Example-based machine translation of the basque language. 2006.

[Stymne and Ahrenberg, 2012] Sara Stymne and Lars Ahrenberg. On the practice of error analysis for machine translation evaluation. In *Proceedings of the 8th Language Resources and Evaluation Conference*, pages 1785–1790, 2012.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Tebbifakhr *et al.*, 2019] Amirhossein Tebbifakhr, Luisa Bentivogli, Matteo Negri, and Marco Turchi. Machine translation for machines: the sentiment classification use case. *arXiv preprint arXiv:1910.00478*, 2019.

[Tiedemann and Scherrer, 2017] Jörg Tiedemann and Yves Scherrer. Neural machine translation with extended context. In *Proceedings of the 3rd Workshop on Discourse in Machine Translation*, 2017.

[Toral and Sánchez-Cartagena, 2017] Antonio Toral and Víctor M Sánchez-Cartagena. A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[Tu *et al.*, 2018] Zhaopeng Tu, Yang Liu, Shuming Shi, and Tong Zhang. Learning to remember translation history with a continuous cache. *Transactions of the Association for Computational Linguistics*, 6:407–420, 2018.

[Uzuner *et al.*, 2011] Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

[Varis and Bojar, 2017] Dusan Varis and Ondřej Bojar. Cuni system for wmt17 automatic post-editing task. In *Proceedings of the 2nd Conference on Machine Translation*, pages 661–666, 2017.

[Vasconcellos, 1989] Muriel Vasconcellos. Cohesion and coherence in the presentation of machine translation products. *Georgetown University Round Table on Languages and Linguistics*, pages 89–105, 1989.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the International Conference on Machine learning*, pages 1096–1103, 2008.

[Wang *et al.*, 2017] Longyue Wang, Zhaopeng Tu, Andy Way, and Qun Liu. Exploiting cross-sentence context for neural machine translation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017.

[Wang *et al.*, 2019] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the International Conference on Learning Representations*, 2019.

[Wei *et al.*, 2016] Qikang Wei, Tao Chen, Ruifeng Xu, Yulan He, and Lin Gui. Disease named entity recognition by combining conditional random fields and bidirectional recurrent neural networks. *Database*, 2016.

[Weizenbaum, 1966] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[Wong and Kit, 2012] Billy Wong and Chunyu Kit. Extending machine translation evaluation metrics with lexical cohesion to document level. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1060–1068, 2012.

[Wu *et al.*, 2015] Yonghui Wu, Jun Xu, Min Jiang, Yaoyun Zhang, and Hua Xu. A study of neural word embeddings for named entity recognition in clinical text. In *AMIA Annual Symposium Proceedings*, volume 2015, page 1326, 2015.

[Wu *et al.*, 2016] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[Xie *et al.*, 2017] Jiaheng Xie, Xiao Liu, and Daniel Dajun Zeng. Mining e-cigarette adverse events in social media using bi-lstm recurrent neural network with word embedding representation. *Journal of the American Medical Informatics Association*, 25(1):72–80, 2017.

[Xiong *et al.*, 2019] Hao Xiong, Zhongjun He, Hua Wu, and Haifeng Wang. Modeling coherence for discourse neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7338–7345, 2019.

[Xu *et al.*, 2019] Weijia Xu, Xing Niu, and Marine Carpuat. Differentiable sampling with flexible reference word order for neural machine translation. *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2019.

[Yang and Hospedales, 2016] Yongxin Yang and Timothy M Hospedales. Trace norm regularised deep multi-task learning. *arXiv preprint arXiv:1606.04038*, 2016.

[Yang *et al.*, 2018] Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. Unsupervised neural machine translation with weight sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[Yang *et al.*, 2019] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the Advances in Neural Information Processing Systems*, 2019.

[Yangarber *et al.*, 2002] Roman Yangarber, Winston Lin, and Ralph Grishman. Unsupervised learning of generalized names. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, 2002.

[Žabokrtskỳ *et al.*, 2008] Zdeněk Žabokrtskỳ, Jan Ptácek, and Petr Pajas. Tectomt: Highly modular mt system with tectogrammatics used as transfer layer. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 167–170, 2008.

[Zaremoodi and Haffari, 2018] Poorya Zaremoodi and Gholamreza Haffari. Neural machine translation for bilingually scarce scenarios: A deep multi-task learning approach. *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2018.

[Zhang and Lapata, 2014] Xingxing Zhang and Mirella Lapata. Chinese poetry generation with recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 670–680, 2014.

[Zhang and Zhu, 2018] Quan-shi Zhang and Song-Chun Zhu. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

[Zhang and Zong, 2016] Jiajun Zhang and Chengqing Zong. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, 2016.

[Zhang *et al.*, 2018] Zhirui Zhang, Shuangzhi Wu, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. Regularizing neural machine translation by target-bidirectional agreement. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[Zhang *et al.*, 2020] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *Proceedings of the International Conference on Learning Representations*, 2020.

[Zhou *et al.*, 2018] Zhong Zhou, Matthias Sperber, and Alex Waibel. Massively parallel cross-lingual learning in low-resource target language translation. In *Proceedings of the Third Conference on Machine Translation*, 2018.

[Zoph *et al.*, 2016] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. *arXiv preprint arXiv:1604.02201*, 2016.