Morphological Zero-Shot Neural Machine Translation

Giulio Zhou

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2018

Abstract

Languages differ from each other in many ways, such as grammar, pronunciation or spelling. However, it is not uncommon to find similarities between them, especially if they belong to the same language family. Previous studies have shown that Neural Machine Translation (NMT) can achieve better performance in a multilingual setting compared to a bilingual one, suggesting their ability to take advantage of shared structures in different languages. The aim of this project is to exploit multilinguality by maximising the intra and interlingual patterns learned by NMT models through a more meaningful morphological word segmentation. This dissertation presents our investigation of the effects of morphological word segmentation for multilingual Zero-Shot NMT models. The models used for our analysis translate between five languages (English, Italian, German, Dutch and Romanian) for a total of twenty translation directions (four of which are Zero-Shot, ie. parallel data for those directions were not provided at training time). In this project, we compare three segmentation strategies: BPE (frequency-based), Morfessor (unsupervised morphological segmentation) and rule-based affix splitter. In addition, we further improve the rule-based algorithm with a modified version of BPE in order to mitigate its oversegmentation problem. The results of our quantitative and qualitative analysis pointed out that morphology helps Zero-Shot NMT models in better capturing morphological variations and grammatical structure compared to BPE-based models, resulting in more fluent (and possibly adequate) translations compared to BPE-based models, as confirmed by the informal human evaluation we conducted. Despite these results, our morphological model did not achieve better performance on automatic evaluation which can be explained in the inability of those metrics to capture characteristics such as fluency.

Acknowledgements

First of all, I would like to thank my supervisor Dr Alexandra Birch for guiding, supporting and motivating me until the very end of this dissertation.

I would also like to thank Olga Becci, Sara Brolli, Jason Fong and Shijie Yao for having actively contributed to this project with meaningful discussions, technical advices and help in proofreading. A special mention goes to Kai Tarafdar who saved this dissertation by providing me with a stash of caffeine.

Finally, thanks to Nico Ring, David Robertson, and all the people who endured all my complaints about how stressed and tired I was for the entire duration of this masters degree.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Giulio Zhou*)

Table of Contents

1 Introduction **1**

2 Background **5**

2.1 Morphology 5

2.1.1 Words and morphology 5

2.1.2 Word formation 6

2.1.3 Language typology 8

2.2 Zero-Shot Neural Machine Translation 9

2.2.1 Neural Machine Translation 9

2.2.2 NMT approaches to Zero-Shot Translation 12

2.3 Word Segmentation Strategies in NMT 15

2.3.1 Words 15

2.3.2 Characters 15

2.3.3 Subwords 16

3 Morphological Segmentation **19**

3.1 Morphological segmentation stategies 19

3.1.1 Rule-based morphological segmentation 19

3.1.2 Unsupervised morphological segmentation 28

3.1.3 Segmentation in practice 28

3.2 Morphological Segmentation Task 30

3.2.1 Method 30

3.2.2 Performance 31

4 Translation Experiments **33**

4.1 Experiments setup 33

4.1.1 Dataset 33

4.1.2 Settings 35

4.2 Experiments results 36
4.2.1 BPE vs Morphologically-Driven Segmentation 36
4.2.2 Low-Resource 39
4.2.3 More settings 42

5 Analysis **45**
5.1 Translation and Morphological Segmentation 45
5.2 Translation Quality with Contrastive Pairs 46
5.2.1 Method 46
5.2.2 Results 47
5.3 Manual Analysis 48
5.3.1 Transliteration Problem 49
5.3.2 Translating tenses 50
5.3.3 Rare words and Out-of-Vocabularies 52
5.4 Informal Human Evaluation 53
5.4.1 Method 54
5.4.2 Results 55
5.5 Discussion on MT automatic evaluation metrics 59

6 Conclusions **61**
6.1 Summary 61
6.2 Future work 62

A Scripts and Additional Material **65**

Bibliography **69**

Chapter 1

Introduction

Let us start with a simple translation test. Given the following *English→Italian* parallel sentences:

I was watching a match→Stavo guardando una partita

She likes to eat apples→Le piace mangiare mele

what is the Italian for "*I was eating an apple*"?

You may feel that there are not enough examples to fully translate the test sentence. It is relatively easy to match *"I was"* with *"Stavo"* and "*an*" with "*una*", but what about "*eating*" and *"apple"*? What if the words in the above training samples were segmented into morphemes (the smallest units of meaning in a word)?

I was watch/ing a match→Stavo guard/ando un/a partit/a

She like/s to eat apple/s→Le piac/e mangi/are mel/e

We can now infer that, in Italian, the infinitive form of a verb is composed by something+"*-are*" and the gerund with something+"*-ando*". Thus, "*eating*" can be translated into "*mangi/ando*". Similarly, we know that the singular form of a noun ends with *"-a"*[1], which applied to "*mel/e*" it becomes "*mel/a*".

The difficulties you might have experienced are not much dissimilar to the ones encountered by a Machine Translation (MT) model. This test showed some potential benefits of using morphological segmented sentences. Unfortunately, as pointed out by Chung et al. (2016), a perfect morphological segmentation algorithm does not exist for any language. In Neural Machine Translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2014), the use of subword units to represent sentences instead of words has become ubiquitous. Currently, the most popular word segmentation strategy is the adapted version of the Byte Pair Encoding (BPE) algorithm proposed by Sennrich et al.

[1] We are purposely ignoring the gender of the word in order to make the example easier to follow.

(2015), an unsupervised and language-independent word segmentation algorithm that does not need a large dataset to be trained. The use of subword units helped to mitigate the rare and out-of-vocabulary problems which affect word-level MT models. Furthermore, due to the reduction in the vocabulary size, subword-based models are much faster to train than word-based ones. Despite these successes, BPE is a frequency-based algorithm and does not provide any linguistic information to the model. While it is possible for BPE to learn some morphological patterns (eg. "*-ing*" is a frequent subword in English, thus it is likely to be learned by BPE), its subwords don't convey morphological information.

Recently more and more studies have been made to analyse the effect of NMT models augmented with additional morphological information. For instance, Dalvi et al. (2017), Tamchyna et al. (2017), Shapiro and Duh (2018) and Passban et al. (2018) all successfully improved translation quality by combining training data with linguistic information such as Part-Of-Speech (POS), Morphological analysis tags or morphologically-trained embeddings. The main drawback of these approaches is the fact that they require additional annotated data or specific linguistic tools which may not be available for the vast majority of the languages.

On the contrary, a simple morphological segmentation can be achieved with only little morphological information or in a completely unsupervised way as shown by Huck et al. (2017b), Ataman et al. (2017) and Banerjee and Bhattacharyya (2018). The results obtained by their studies demonstrate that even without a perfect morphological segmentation algorithm, it is possible to improve sensibly the performance of the NMT models. In addition, the morphological NMT model proposed by Huck et al. (2017b), despite obtaining a lower BLEU score compared to other models, resulted to be the state-of-the-art model in terms of fluency and adequacy for the *English→German* language pair.

In this project, we will evaluate the impact of morphological segmentation on Zero-Shot NMT. Zero-Shot translation can be considered as a harder problem compared to classic bilingual translation due to its extreme low-resource condition. In fact, as defined by Johnson et al. (2016), Zero-Shot translation is the ability of a model to translate between language pairs for which no direct parallel data was provided. The intuition behind this project is that morphological information would be more effective on Multilingual Zero-Shot translation compared to bilingual translation. This hypothesis is backed by the results obtained in other studies. For instance, Shapiro and Duh (2018) showed that translation in low-resource conditions benefits from morphological

information, while Banerjee and Bhattacharyya (2018) discovered that models based on morphological segmentation benefit the most when translating between morphologically similar languages. This suggests that NMT models are capable of learning linguistic patterns in languages, and more importantly between languages. As a result, we believe that these benefits brought by morphological segmentation are amplified when used in multilingual settings since similar patterns are shared between the various languages (unless they are all unrelated) and that these learned patterns will affect positively the translation quality of Zero-Shot directions.

Given the above, the objective of this project is to provide an extensive in-depth analysis of Zero-Shot NMT models which rely on morphologically-driven segmentations instead of solely on BPE. We will thus try to empirically answer the following questions:

- Does morphological segmentation improve the performance of Zero-Shot NMT models?
- Are the improvements/loss greater compared to non-Zero-Shot models?
- What is the correlation between morphology segmentation accuracy and translation quality?
- Are morphological-driven models more capable of dealing with linguistic phenomena (eg. inflections, subject-verb agreement etc...) compared to BPE?

To answer these questions, we compare the performance of three types of word segmentation strategies (BPE, rule-based (as in Huck et al. (2017b)) and unsupervised morphological segmentation (Virpioja et al., 2013)) on the IWSLT'17 Zero-Shot translation task (Mauro et al., 2017).

Our contribution is three-fold:

- We adapt the BPE algorithm for a new task: merging subwords together. This algorithm (which we call *mergeBPE*) successfully mitigates the oversegmentation problem that affects imperfect morphological segmentation strategies;
- Analysis with contrastive pairs showed that morphological-driven models are more effective in disambiguating grammatical errors (eg. agreements, polarity, compounding etc...) with the only weakness being transliteration. In addition, qualitative analyses suggest that morphological translations are possibly more fluent/adequate despite obtaining lower BLEU scores;

- For the translation experiments, our morphological models did not outperform BPE-based model on automatic evaluation metrics such as BLEU and chrF3 (except for a few translation directions). We conjecture that this result is due to two factors: the inability of automatic scores in capturing translation quality and ambiguity brought by non-perfect morphological segmentation.

The rest of this dissertation is structured as follows: Chapter 2 will provide background information on Morphology, NMT architectures and the application of word segmentation strategies, Chapter 3 will describe the implementation of the morphological segmentation that will be used for the translation experiments and a short analysis of how morphological those algorithm actually are, Chapter 4 will provide the results and comparison of the various morphological models on translation tasks while Chapter 5 will analyse the performance of the models under a different perspective, providing quantitative and qualitative analysis, to then conclude with a summary of this project and possible future work in Chapter 6.

Chapter 2

Background

2.1 Morphology

2.1.1 Words and morphology

Traditionally, morphology is defined as the study of the internal structure of words and their formation. However, the definition of a word itself is ambiguous. For instance, how many words are contained in the following quote?

> “ It is impossible to live without failing at something, unless you live so cautiously that you might as well not have lived at all - in which case, you fail by default. ”
>
> *J.K. Rowling*

From a certain point of view, we can say that there are thirty-one words. In fact, there are thirty-one distinct elements in the sentence separated by white spaces. However, there are multiple occurrences of *“live”, “you”* and *“at”* which may be counted only once each. Furthermore, we might want to count only words that appear in the dictionary but we might notice that there is an entry for *“fail”* but not *“failing”*, and similarly, for *“live”* and not *“lived”*.

Given this ambiguity, it is necessary to use special terms to differentiate each case. Formally, in the quote above there are thirty-one **word tokens**, twenty-eight **word types** (distinct tokens) and twenty-six **lexemes** (unit of lexical meaning).

Despite the definitions above, it is still unclear what a “real” word is. Is “*aa*” a word? What about ‘*unyellow*”? According to the *Oxford English Dictionary* “*aa*” is a volcanic rock, while it does not have an entry for “*unyellow*”. However, *“unyellow”*

is a plausible English word which can mean "not yellow". We are able to infer this meaning by recognising smaller units in the word such as "*un*" and "*yellow*" called **morphemes** which can be defined as "the smallest unit of meaning in a language" (Bender, 2013).

From the examples above, we can see that not all the morphemes are the same, in fact, while *"yellow"* is a stand-alone word, "*un*" never appears by itself in the English language and it needs to be attached to other morphemes. The first category are thus called **free** morphemes while the second **bound** morphemes. A free morpheme is also called the root of a word which combined with other morphemes forms new lexemes called **derived** or **complex** words.

2.1.2 Word formation

Word formation is the process of combining morphemes to create new words. However, it is not possible to freely merge morphemes. Figure 2.1 shows the wug test proposed by Berko (1958). The test consists of identifying the plural form of the word "*wug*". Despite not knowing what a wug might be, we can agree that the plural form of *"wug"* is *"wugs"*. Why not just *"wug"*, *"wugges"* or something else? This is because each language has its own **word formation rules** that allow us to create meaningful and well-formed words.

Figure 2.1: The Wug Test (Berko, 1958)

2.1.2.1 Affixation

The most common word formation technique is affixation, where a bound morpheme (**affix**) is combined with a free morpheme. Affixes are categorised by their type and

position[1]:

- **Suffix**: placed at the end of the word (eg. "*fail*"→"*fail**ing***");
- **Prefix**: placed at the start of the word (eg. "*activate*"→"***de**activate*");
- **Infix**: placed in the middle of the word (eg. in Turkish "*geliyorum*" → "*gel**m**iyorum*");
- **Circumfix**: composed of two parts which are placed at the start and end of the word (eg. the German past participle is obtained with the morpheme "*ge-...-t*");
- **Transfix**: discontinuous affix (pattern morphology) (eg. in Arabic "*ktb*" → *k**a**t**a**b**a***);

As we have anticipated above, it is not possible to attach any affix to a free morpheme. Each language has set word formation rules expressed in phonological, meaning and part of speech (POS) requirements. For example the word "*unhappiness*" is composed of the morphemes "*un-*", "*happy*" and "*-ness*". Both affixes have POS requirements, in fact, "*un-*" can precede a verb or an adjective but not a noun, while "*-ness*" follows only adjectives. We apply the prefix "*un-*" to an adjective in order to make it negative but it cannot be applied to a word that's already negative (eg. *"sad"*). Similarly, "*un-*" can be used with a verb, but not if it's an irreversible action (eg. *"explode"*).

The process of affixation does not simply consist in the attachment of an affix to another morpheme. As we can see from the word "*unhappiness*", the morpheme "happy" changed to "*happi*" when combined with "*-ness*". In fact, it is common that letters in the morphemes are added, removed or modified due to spelling rules.

2.1.2.2 Compounding and reduplication

Compounding is the creation of new words by combining one or more roots. Examples of compounds in English are: "*windmill*", "*greenhouse*", "*bedroom*", "*firefighter*", etc...

Although the various compound segments are semantically independent, in a compound there is a **head** that gives both the POS and the main semantic denotation of the compound. In English, the head is generally the most right morpheme (eg. in greenhouse the head is house), for this reason, English is called right-headed. Other languages such as French have the heads on the left and thus called left-headed.

[1] Some examples are from (Bender, 2013).

Similarly to compounding, reduplication is the concatenation of the same morpheme. For instance in Chinese reduplication is used with verbs to express "a little" (eg. "*kàn*" (看 , to look) → "*kànkan*" (看看, to take a little look)).

2.1.2.3 Derivational vs. inflectional morphology

Above we have defined word formation as the process of creating new words by combining multiple morphemes. However, we can distinguish two types of word formation: derivational and inflectional.

The main difference between derivational and inflectional morphology is that the first creates new lexemes while the second produces a word with the same lexeme of the uninflected word. More specifically, inflectional morphology modifies lexemes in order to fit them into various syntactic contexts (Lieber, 2017) by changing its number, gender, tense, etc...

Although derivational morphology might change the syntactic category of the word (eg. "*happy*" (Adjective) → "*happiness*" (Noun)), it is also considered derivational morphology when the meaning of the original word change considerably (eg. "*happy*" (Adjective) → "*unhappy*" (Adjective)).

2.1.3 Language typology

Based on their morphology, languages can be classified as agglutinative, fusional, isolating or polysynthetic (Lieber, 2015).

- **Agglutinative**: words can be easily segmented into morphemes, each of them carrying only one piece of meaning (eg. Turkish);
- **Fusional**: compared to agglutinative languages, it is harder to distinguish the boundaries of the morphemes. In addition, a single morpheme may carry multiple meanings (eg. in Italian suffixes for the plural form contain information of the gender as well. "*ragazz-a*" is singular feminine while "*ragazz-i*" is plural masculine);
- **Isolating**: languages with a low level of morphological inflection (eg. Chinese);
- **Polysynthetic**: languages with words containing many morphemes (eg. Greenlandic (Fortescue, 1984).)

2.2 Zero-Shot Neural Machine Translation

2.2.1 Neural Machine Translation

Neural Machine Translation (NMT) is an approach to Machine Translation (MT) which uses neural networks as core components. NMT has proven itself to be the most promising paradigm in the field, achieving state-of-the-art performance in an end-to-end fashion (Bojar et al., 2016, 2017). Currently, the most widely adopted architecture in NMT is the Encoder-Decoder Recurrent Neural Network (RNN) with attention mechanism proposed by Bahdanau et al. (2014). Despite variations of this model have been proposed, the core mechanism has remained the same over the past few years. In this section, we will describe the main components of such architecture as shown in Figure 2.2.

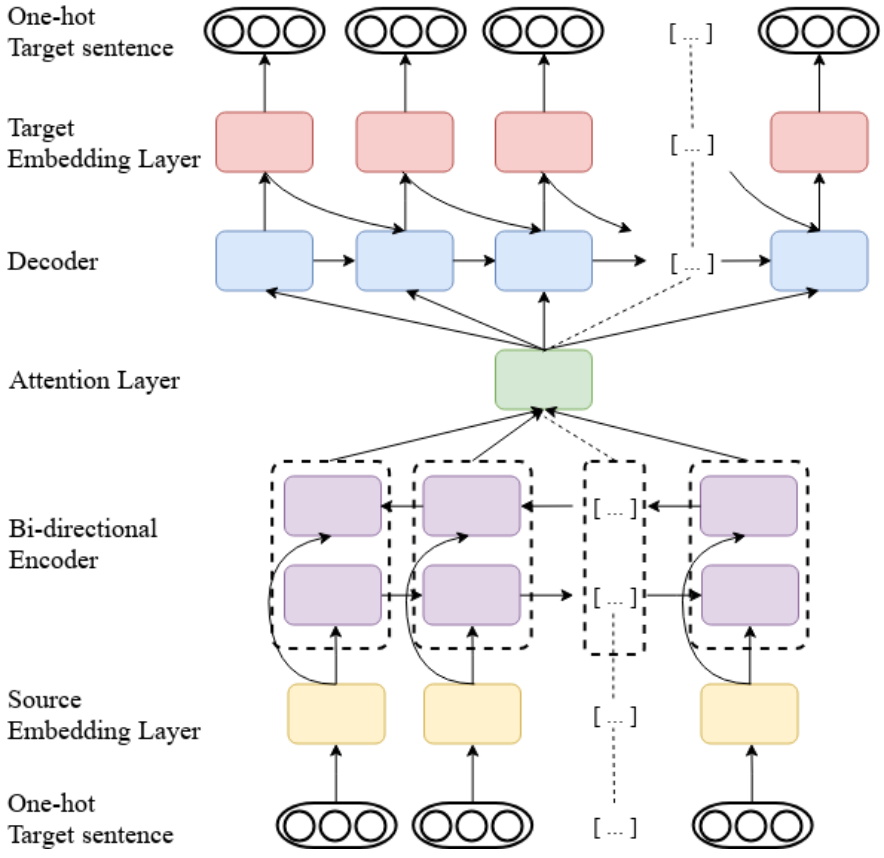


Figure 2.2: Bi-directional Encoder-Decoder RNN with attention mechanism as proposed by Bahdanau et al. (2014).

An RNN is a generalisation of a feed-forward neural network (FFNN) which takes a sequence $\mathbf{x} = (x_1, \ldots, x_T)$ as input and returns a sequence of the same length $\mathbf{y} =$

$(y_1, \ldots, y_T)$. When computing each element of the sequence (time step), the activation unit takes as input not only the input itself but also the hidden state of the previous time step: $\mathbf{h}_t = f(\mathbf{h}_{t-1}, x_t)$, where f is an activation function (or a gated unit as described later in this section). Since in MT input and output have not the same length, Sutskever et al. (2014) proposed the Encoder-Decoder architecture which is composed of two RNNs: an Encoder that converts the input sequence into hidden states, and a Decoder that generates words in the target language. Compared to Sutskever et al. (2014), Bahdanau et al. (2014) introduced three improvements:

Attention mechanism: instead of passing only the last hidden state of the Encoder to the Decoder, at each time step the Decoder takes as input a context vector c_t which is a weighted sum of all the hidden states produced by the Encoder (thus one for each input unit) as shown in the following equations.

$$c_t = \sum_j a_{tj} \mathbf{h}_j \qquad a_{tj} = \frac{\exp(att(\mathbf{s}_t, \mathbf{h}_j))}{\sum_k \exp(att(\mathbf{s}_t, \mathbf{h}_k))}$$

where $\mathbf{h}_i$ is the hidden state of the i^{th} input word and $att(\mathbf{s}_t, \mathbf{h}_j)$ a function that compute an alignment score between the hidden states (Luong et al. (2015) provide a list of possible scoring functions).

Gated units: vanilla RNNs are known to perform poorly with long-distance dependencies because information is overwritten at each time step. However, this problem is mitigated by using gated units such as Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) or Gated Recurrent Unit (GRU) (Cho et al., 2014) cells instead of just an activation function.

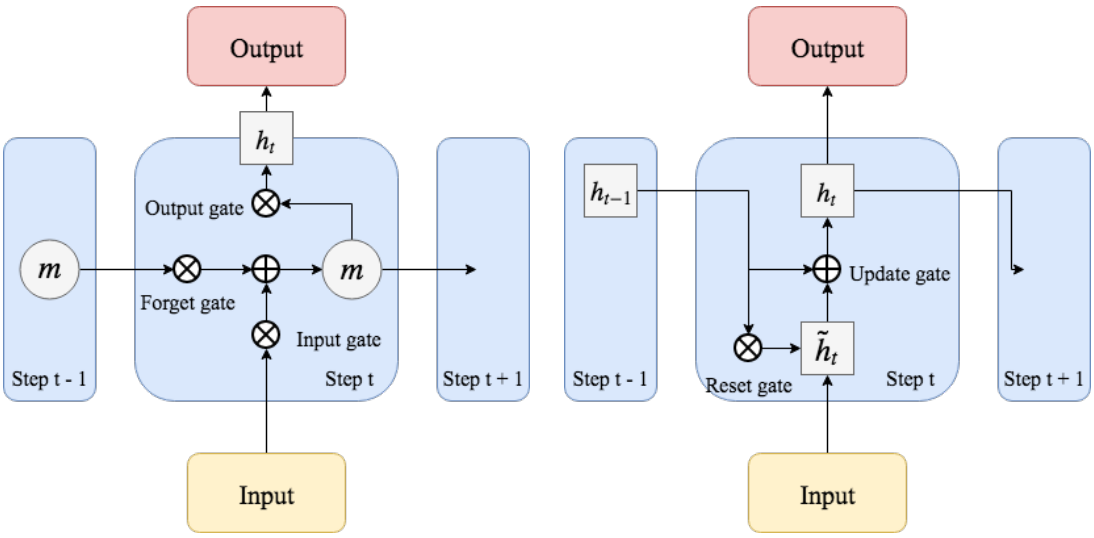
Figure 2.3 shows the schematised[2] version of these cells as in Koehn (2017). An LSTM cell is composed of a memory state $\mathbf{m}$ and three gates (input, output and forget) which perform read, write and reset operations. As the name *gate* suggests, the purpose of these operations is to control the amount of information to retain from the input, current and previous time step memory cell. The memory cell and output hidden state are computed as follow:

$$\mathbf{m}_t = gate_{input} \times \mathbf{x}_t + gate_{forget} \times \mathbf{m}_{t-1}$$

$$\mathbf{h}_t = f(gate_{output} \times \mathbf{m}_t)$$

where $\times$ and $+$ are point-wise operations, and f an activation function.

[2]For this project we are not interested in the details of the implementations. For this reason, for the full version and actual Maths, please refer to the original papers.



(a) LSTM (b) GRU

Figure 2.3: Gated units. m is the memory cell, h_t is the produced hidden state at time t, $\tilde{h}_t$ is the provisional hidden state computed by the GRU cell. The input and outpul cells represent other LSTM/GRU cells or embedding layers.

GRU cells are composed of two gates (the update and reset gates) but they do not have an explicit memory state. Instead, an internal hidden state is computed by applying the reset gate to the hidden state of the previous time step and the input. The update gate produce the actual output of the cell by combining the newly computed internal hidden state and the one from the previous time step.

$$\widetilde{\mathbf{h}}_t = \phi(W\mathbf{x}_t + U(gate_{reset} \times \mathbf{h}_{t-1}))$$

$$\mathbf{h}_t = gate_{update} \times \mathbf{h}_{t-1} + (1 - gate_{update}) \times \widetilde{\mathbf{h}}_t$$

where W and U are weight matrices, and ϕ an activation function.

Bi-directional RNN: A regular RNN takes one input at a time from left to right. With this architecture, a single hidden state incorporates only information from the previous word. In order to contextualise the word among all the surrounding words, a bi-directional RNN scans the input sequence both from left to right and right to left. This architecture is composed of two separate RNNs as shown in Figure 2.2. The hidden state of each word is thus a concatenation of the hidden states produced by the forward and backward RNN $\mathbf{h}_i = [\overrightarrow{\mathbf{h}_i}, \overleftarrow{\mathbf{h}_i}]$.

2.2.2 NMT approaches to Zero-Shot Translation

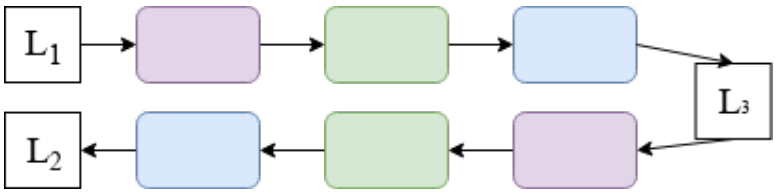
In Machine Learning, the term Zero-Shot is used to define the condition of learning new concepts through textual description (Romera-Paredes and Torr, 2015) (eg. classification of out-of-category images through their description). Instead, in Machine Translation, Zero-Shot took a slightly different connotation: Johnson et al. (2016) defined Zero-Shot as the ability of a model to translate between language pairs for which no parallel data was provided at training time, which differs from Zero-Resource translation where an additional fine-tuning step with synthetic parallel data is required.

2.2.2.1 Enabling Zero-Shot

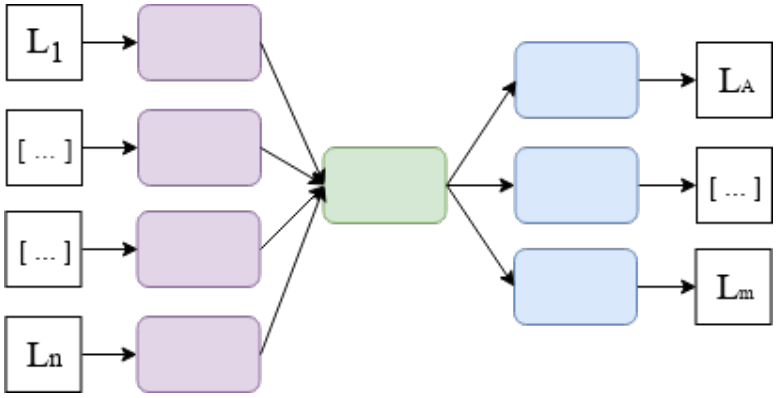
NMT approaches to Zero-Shot Translation can be categorised into three main groups based on the number of NMT components (embeddings, Encoder, Decoder, attention) shared by the various languages as shown in Figure 5.3:

No sharing: This group can also be called *pivot-based* models. In this type of models, no parameter is shared between translation directions. Zero-Shot is achieved by explicitly bridging the source and target language with a third one called *pivot*. For this reason, two independent models are trained: source→pivot and pivot→target. The main problem with this approach is error propagation since the second model relies on the quality of the translations of the first model. To mitigate this problem, a solution was proposed by Cheng et al. (2017), which consists in connecting the first Decoder to the second Encoder to then jointly train the two models without producing explicit intermediate translations.

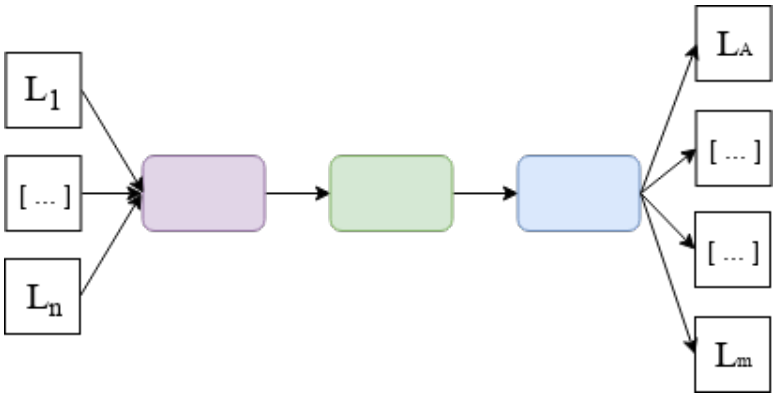
Partial sharing: In order to achieve Zero-Shot translation, it is not possible to avoid the use of subsidiary languages. However, a multilingual model with shared parameters between languages can enable it without the *two-step* translation described above. For instance, Firat et al. (2016) proposed an architecture where each language pair has its own Encoder and Decoder, however, the attention mechanism was shared between them all (multi-way multilingual NMT) making *one-to-many*, *many-to-one*, *many-to-many* translations possible. Despite improving the translation quality of the bilingual model, the Zero-Shot results were poor. Another architecture with partial parameter sharing was proposed by Pham et al. (2017), which, on the contrary of Firat et al. (2016), shared one Encoder for all the source languages and one Decoder for the target languages while maintaining explicit one attention mechanism for each language pair. Although this model seemed to have a better performance compared to



(a) no sharing (Cheng et al., 2017)



(b) partial sharing (Firat et al., 2016)



(c) all sharing (Johnson et al., 2016)

Figure 2.4: Three approaches to enabling Zero-Shot translation in NMT. Encoders in purple, Attention layers in green and Decoders in blue.

multilingual models with no parameter sharing (especially on low-resource condition), its effectiveness for Zero-Shot translation is not clear.

All sharing: as the name suggests, all the parameters are shared between the language pairs, which means that the model is composed of one Encoder, one Decoder and one attention mechanism. This type of multilingual model was proposed by Johnson et al. (2016). In order to enable Zero-Shot translation, they proposed a technique called *target forcing* which consists in concatenating a special token at the beginning of each sentence in order to specify which language translate the sentence into. The main advantage of this approach is that it does not involve any modification to the architec-

ture of the NMT model, permitting to scale the number of the language more easily compared to other techniques such as Firat et al. (2016). In addition, by effectively sharing the parameters between languages, Johnson et al. (2016) have demonstrated that it improves the performance of low-resource language pairs.

2.2.2.2 Recent improvements

Over the past year, many authors attempted to improve the performance of Zero-Shot NMT models by extending the methods described above.

Language independent embeddings: Gu et al. (2018) proposed a method to improve the word embeddings of the source languages by projecting the words into a shared word embedding space called *Universal Lexical Representation*, and a second component called *Mixture of Language Expert*s needed to map these shared representations to the various language structures. Similarly, Lu et al. (2018) generates language independent word embeddings through an *interlingua* component placed between the Encoders and the Decoders (architecture similar to Firat et al. (2016)).

Iterative learning: Sestorain et al. (2018) combined Zero-Shot with dual learning (use of monolingual data and reinforcement learning to incrementally improve translation quality (He et al., 2016)) while Lakew et al. (2017) use self-generated data and an iterative learning process that "leverages a duality of translations directly generated by the system for the zero-shot directions".

Improved data/vocabulary: Ha et al. (2017) address the language bias problem (ie. translations in the wrong target language) in multilingual systems by disabling the vocabulary entries not related to the target language, or by enriching the training corpora with *target→target* data. España-Bonet and van Genabith (2017) have instead improved translation quality by augmenting the data with linguistic and semantic annotations (ie. POS, Lemma, Stem, topic etc...). Despite the hypothesis of their work are similar to the ones of this project (linguistic factors being beneficial to Zero-Shot NMT), their approach differs substantially from ours. In fact, their method heavily relies on additional linguistic tools and more annotated data, while we inject linguistic information indirectly into the model through a morphological segmentation.

Different architecture: While Encoder-Decoder RNN with attention mechanism is still the most wildly adopted architecture for NMT, other architectures based on other types of neural network have been developed. The most recent one is *Transformer* proposed by Vaswani et al. (2017). Instead of adopting recurrent layers, the Transformer uses feedforward *self-attention* layers. In addition, to obtain comparable

if not better performance than traditional NMT models, it is also considerably faster to train. This architecture was successfully used to enable Zero-Shot translation by Dabre et al. (2017). Their model is currently the state-of-the-art for Zero-Shot translation, in fact, in a multilingual setting, it didn't suffer any loss in terms of BLEU score when removing the Zero-Shot directions in the training corpus.

2.3 Word Segmentation Strategies in NMT

NMT belongs to the category of sequence-to-sequence learning. In fact, it learns to map sentences in a source language to a sentence in a target language. Since it is not possible to feed sentences as they are, we need to convert them into a representation suitable for an NMT architecture. NMT models take as input and output *one-hot* vectors (sequences of zeros with a single element set to one) with the dimension of the vocabulary as the size of the vector. However, the NMT model is independent of the entries of the vocabulary. For this reason, we can represent a sentence in various forms, like a sequence of words, characters, subwords or any other symbol encodings.

2.3.1 Words

The vast majority of SMT models were based on words as the basic unit. The popularity of this representation method was because of two reasons: a word naturally expresses a unique meaning (ie. even if different words share the same lexeme, their underlying meanings differ), smaller units result in longer sequences, which results in vanishing gradient and more difficulties in handling long-distance dependencies.

In NMT, word-level representation was quickly discovered to be inadequate. Due to the high computational complexity of neural models and the size of the model being directly proportional to the vocabulary size, an NMT dictionary is generally composed of only 30 000 - 50 000 high-frequency words (Sennrich et al., 2015) while all the remaining words were represented by a single "UNK" token. As a consequence, word-level NMT models were not effective in translating rare or unseen words.

2.3.2 Characters

The benefits and problems of character-level models were thoroughly described by Chung et al. (2016) and Lee et al. (2016).

The main advantage of a character-level model is that, without having to explicitly inject linguistic knowledge into the model, it is capable of dealing with morphological variations, infrequent and unseen words. However, there are several shortcomings. First of all, despite the fact that a character-level model is able to learn recurrent patterns in languages, it requires more data compared to models that use bigger units. This is because it has to learn how to build every single word from scratch. Secondly, as we mentioned above, it suffers when dealing with long-distance dependencies, even when using memory units like LSTM or GRU. Lastly, it is hard to map sequences of characters to a continuous representation, to overcome this problem additional layers are required (eg. Lee et al. (2016) uses convolutional layers before the embedding layer).

2.3.3 Subwords

In addition to the representation methods presented above, a sentence can be represented as a sequence of subwords. We can define a subword as a word segment. Although character-level models can be considered subword-level models, they are generally excluded from this category. The most naive way to achieve subwords is through n-gram segmentation, however, more complex segmentation techniques have been developed like frequency-based or linguistic-based segmentations.

2.3.3.1 BPE

The most widely used subwords in NMT are BPE segments. In order to overcome the rare/unseen words problem, Sennrich et al. (2015) proposed the use of subword units and new word segmentation strategy: an adaptation of the Byte Pair Encoding (BPE) algorithm that learns to combine characters/subwords instead of bytes. Like the original BPE algorithm, the one developed by Sennrich et al. (2015) is a frequency-based algorithm. The algorithm works in two phases: learning and segmenting.

The learning process is iterative, it takes as input a training corpus and the number of iterations (also called merge operations) and returns a BPE model composed of a list of subword pairs ordered by frequency. The model starts with the training data segmented into characters. At each iteration, the most frequent subword pair (at first, character pair) is found. The subwords in the pair are combined together to form a new unique symbol and it is replaced in the training data. For instance, given the corpus “ABC ABABC”, it is first divided into “A B C — A B A B C”, the most frequent

subword pair is “A B”, thus they are combined to form “AB”, the resulting corpus would be “AB C — AB AB C”. At the second iteration, the frequency of the pairs are recalculated and “AB C” are combined. At segmentation time, a corpus and a BPE model is taken as input and while the output is the segmented corpus. One word is processed at a time according to the frequency pairs in the model. In addition to the actual segmentation, each subword is delimited by a separator (in Sennrich et al. (2015) they add “@@” at the end of the subword). So, given “ABCAAB” and the model trained above, the resulting segmented word would be “ABC@@ A@@ AB”

BPE as word segmentation strategy has several advantages. First of all, it is an unsupervised, language independent and it does not require a large training corpus in order to learn the frequent pairs, which make it extremely versatile and easily applicable to many languages. Secondly, it effectively reduces the vocabulary size of a model and segments out-of-vocabulary words into seen subwords (in our example “ABCAAB” was an unseen word). BPE obtained immediately impressive results, and it was rapidly adopted by most NMT models (Bojar et al., 2017).

2.3.3.2 Morphological Segmentation

As pointed out by Chung et al. (2016), the perfect word segmentation algorithm would be one that is able to perfectly segment sequences into lexemes and morphemes. However, at the current time, there is no such algorithm for any language, due to the fact that morpheme boundaries are not always obvious as have discussed in Section 2.1.

Despite various studies that have been conducted to analyse the morphological level of NMT model and the positive effect of morphological information injected into such models (Belinkov et al., 2017; Dalvi et al., 2017; Tamchyna et al., 2017; Shapiro and Duh, 2018; Passban et al., 2018), to the best of our knowledge, only Huck et al. (2017b), Ataman et al. (2017) and Banerjee and Bhattacharyya (2018) implemented a morphological-driven word segmentation for NMT.

Huck et al. (2017b) performed experiments on the *English→German* translation direction. Morphological segmentation is applied before BPE on the target-side. Since German is an extremely morphologically-rich language, a morphological segmentation was used in order to help the decoder generate inflections and compounds. The segmentation algorithm proposed is composed of three steps (plus BPE). The first two are suffix and prefix segmentation through simple rule-based algorithms[3], while the

[3]More implementation details will be provided in Section 3.1.1.

third is compound splitting as in Koehn and Knight (2003). Compared to their baseline, a morphological-driven target side slightly improved the translation quality of the model, even when only the suffix splitter was applied. In addition, their submission for the WMT17 shared task (Huck et al., 2017a), despite not obtaining the highest BLEU score, it was ranked first by human evaluation (Bojar et al., 2017).

Contrary to Huck et al. (2017b), Ataman et al. (2017) applied a morphological segmentation strategy on the source-side (without BPE on top of it) while maintaining BPE on the target side. The language pair involved is *Turkish→English*. Two different approaches have been tested: supervised (a combination of morphological analysis tools for the Turkish language that leaves the root of the word and its inflection in terms of the role of the suffix) and unsupervised (Morfessor *Flatcat* (Grönroos et al., 2014)) morphological segmentation. Both morphological-based models outperformed the baseline considerably (≈+2 1 BLEU score). The unsupervised model performed slightly better than the supervised one. As suggested byAtaman et al. (2017), this result may be related to the loss caused by the morphological analysis. In fact, the supervised method was not an exact morphological segmentation.

Similarly to Ataman et al. (2017), Banerjee and Bhattacharyya (2018) uses Morphessor *Flatcat* as a morphological segmentation strategy, however it is used in conjunction with BPE and on both source and target-side. A further difference from Huck et al. (2017b) is that while Huck et al. (2017b) uses different separators between the various segmentation step, Banerjee and Bhattacharyya (2018) converts all the separators into a single type (“@@”). They studied the performance of this segmentation on *English→Hindi*,*English→Bengali* and *Bengali→Hindi*. For all the directions, the morphological models outperformed the BPE baseline. In addition Banerjee and Bhattacharyya (2018) discovered that the effectiveness of their approach was higher when the languages involved are lexically close.

Chapter 3

Morphological Segmentation

The objective of morphological segmentation is to break words into morphemes. In this section, we present the morphological segmentation strategies that will be used for the Zero-Shot translation experiments. In the first part, we will describe the various segmentation techniques and their implementation. In the second part, we will evaluate them on a morphological segmentation task, focusing on the analysis of the strengths and weaknesses of each method.

3.1 Morphological segmentation stategies

As discussed in Section 2.1, Morphology can be expressed in a variety of ways. Although more complex forms of morphological variations may frequently occur in certain language families (eg. pattern morphology in Semitic languages (Arad, 2006)), the vast majority of them occur in the form of suffixes, prefixes or compounding.

We hereby describe two methods for individuating and splitting morphemes.

3.1.1 Rule-based morphological segmentation

One of the possible approaches to morphological segmentation is rule-based segmentation. The main advantage of this type of algorithm is that it is simple to implement and it only requires someone who has a sufficient knowledge of the linguistic phenomena that occur in the target language. This might be extremely useful for implementing a tool for Low-Resource languages for which no tagged data are available, or even no digital data at all. As a drawback, languages are complex, especially those with blurred morpheme boundaries or irregular forms such as fusional languages. Furthermore, a

rule-based algorithm is language-specific and it is not generalisable.

3.1.1.1 Suffix Splitting

For the rule-base suffix segmenter, we developed two different types of algorithm. An adapted version of the NLTK Snowball stemming algorithm[1] for morphological segmentation as proposed by Huck et al. (2017b) and one from scratch based on our linguistic knowledge of the languages (only for English and Italian). The main reason that led us to implement the second type of algorithm is that by analysing the NLTK Snowball stemming algorithm we discovered that it considers only a limited number of suffixes while ignoring completely the large number of *suffixoids* (morphemes that are in between free and bound morphemes) that occur in many languages. Moreover, we wanted to test the complexity of implementing a rule-based algorithm from scratch since, currently, only 15 languages are supported by the NLTK toolkit.

3.1.1.1.1 NLTK based algorithm The NLTK Snowball stemming algorithm uses some language-specific heuristics to cut the suffixes (Huck et al., 2017b). According to the notes in the source code, the algorithm individuates special regions in the word called R1 and R2[2]:

- R1: is the region after the first consonant following a vowel (eg. in “beautiful” “t” is the first consonant that follows a vowel, thus, the R1 is “iful”);
- R2: same as R1 but it is applied to the R1 region of a word and not the word itself (eg. since the R1 region of “beautiful” is “iful”, the R2 of “beautiful” is “ul” since “f” is the first consonant following a vowel in the R1 region).

In addition to these two regions, variations or additional regions are defined to adapt the algorithm to specific languages. For instance, for Italian, Spanish, Portuguese and Romanian, an RV region is defined as the region after the first vowel following a consonant if in second or third position, or the region after the third letter.

Based on the language, the algorithm passes the input word through a number of steps that usually match with the suffix category (eg. tense inflections, derivational suffixes, pronouns, etc...). For each step, it checks whether one of the previously described regions ends with a suffix in the respective list and drops it if found. In ad-

[1] https://www.nltk.org/_modules/nltk/stem/snowball.html
[2] http://snowball.tartarus.org/texts/r1r2.html

dition, multiple language-specific controls are made within each step in order to deal with morphophonetical variations.

To convert the stemming algorithm into a segmenter algorithm, we modified the source code so that the suffixes found are saved and returned. In addition, minor changes were needed to deal with the *suffix_replace()* method in the algorithm which, as the name suggests, replace the suffix with some other sequence of characters instead of just dropping it. For instance, in Dutch the suffix "heden" is replaced with "heid" and it drops only if other conditions are met in other steps of the algorithm, or in Italian "ar", "er" and "ir" are replaced with "are", "ere" and "ire".

With this technique we produced morphological segmenter for German, Dutch and Romanian.

3.1.1.1.2 Suffix segmenter from scratch

Suffix segmenters for Italian and English were developed from scratch through an iterative process. The core of the segmeter is based on the following pseudocode:

Algorithm 1: Suffix segmenter core pseudocode

```
input: word
result ← " "
if word in stopwords or len(word)< 4 then
    return word
end if
while True do
    for suffix in suffixList do
        if word.endswith(suffix) then
            result ← suffix + " " + result
            word ← word[: −len(suffix)]
        end if
    end for
    if unchanged then
        break
    end if
end while
return word + " " + result
```

This procedure is not dissimilar to the one used in the NLTK Snowball stemming algorithm, in fact, the main idea is to iteratively check whether the word ends with

a suffix and split it. However we do not use R1, R2 or RV regions. To fasten the development process, the lists of suffixes were retrieved from online resources[3]. The algorithms were extended with a 'try&fix' method by refining and adding special cases. For instance, in English the plural form is achieved by appending "s" at the of the word, however, many words in the language end with such letter, for this reason, it is not possible to indiscriminately segment it.

English Suffix Segmenter

Algorithm 2: English Suffix Segmenter

return ***if*** *the input is a stopword*
split adverb suffixes (eg. "-ly", "-ward", ...)
split plural forms ("-s", "-es", "-ies")
    ignore if words end with "-ss", "-us", "-is" or "-os"
while *result modified* **do**
    split noun/adjective suffixes (eg. "-ful", "-ish", "-ness", ...)
        split "a" or "e" before "-nce", "-s", "-t","-x" before "-ion", "-i" before "-ble" or "-al"
    split phonetical vowels and double consonants
end while
split "-ed" or "-ing"
split phonetical vowels and double consonants
split verb suffixes (eg. "-ify", "-ent", "-ate", ...)
    ignore if words end with "-ment"
return *result*

Algorthm 2 outlines the steps made by the English suffix segmenter in order to individuate the suffixes. As we can see, compared to Algorithm 1, several suffix categories are segmented only once and in a specific order.

We first check whether the word is an adverb since adverbial suffixes occur only after all the other suffixes. After splitting the plural inflections, we iteratively segment all the noun and adjective suffixes until no modifications are made. This is because, theoretically, it is possible to have an infinite combination of adjective-to-noun, noun-to-adjective and noun-to-noun derivational affixes. The algorithm finishes by segmenting the tense inflections and the noun/adjective-to-verb suffixes.

[3]English-suffixes→https://dictionary.cambridge.org/grammar/british-grammar/word-formation/suffixes
Italian-suffixes→https://it.wikipedia.org/wiki/Suffissi_della_lingua_italiana

For all the suffixes, we tried to make them as much "shared" as possible. For example, the suffixes "-able" and "-ible" are technically distinct suffixes, although their meanings are the same. For this reason we consider "-ble" as morpheme instead of keeping both versions. As we can see from Algorithm 2, we separate vowels and double consonants between morphemes. This is to both handle variations of the same suffix and letters modified for phonetic reasons (eg. double consonant → "begin**n**ing", vowel deletion → "relat**e**"/"relation", modification → "pretty"/"pretti**e**st").

Italian Suffix Segmenter

Algorithm 3: Italian Suffix Segmenter

return ***if*** *the input is a stopword*
split adverb suffixes (eg. "-mente")
  split "a" before "-mente"
if *word* ***ends with*** *clitic pronouns* **then**
  split clitic pronouns (eg. "gli", "ci", "le", ...)
    ignore if verb is not imperative, gerund or infinitive
    separate compound clitic pronouns (eg. "gliele")
else
  if *word* ***ends with*** *verb conjugation* **then**
    split conjugation (eg. "essimo", "erei", "iro", ...)
      split "e" after "ar", "er", "ir"
    split noun-to-verb suffixes ("ific", "izz", "eggi")
  end if
end if
split phonetical vowels and "h"
split suffixoids (eg. "grafi", "logi", "tomi", ...)
while *result modified* **do**
  split noun/adjective suffixes (eg. "acci", "evol", "ism", ...)
end while
split phonetical vowels and "h"
return *result*

The functionality of the Italian suffix segmenter is similar to the English one, however, being a morphological-richer language compared to English, more language-specific cases were needed. As for the English segmenter, the algorithm starts by splitting adverbial suffixes.

One of the particularities of the Italian Segmenter is how it deals with verbs. Italian,

as for other Romance languages, has an extremely complex verb system. In fact, an Italian verb has 21 tenses with most of them having 6 inflections ($1^{st}, 2^{nd}, 3^{rd}$-person singular/plural) which have to multiplied by 3 verb conjugations ($1^{st}, 2^{nd}, 3^{rd}$ conjugation which end with "-are", "-ere" and "-ire" respectively) (Pizzuto and Caselli, 1994). In addition to verb tenses, we found postverbal clitic pronouns (Casadio and Lambek, 2001) problematic to segment. Postverbal clitic pronouns are monosyllabic particles that are attached at the end of the verb in order to express accusative, dative or locative information. For instance, "andiamoci" is formed of "andiamo"→"let's go", "ci"→"there", or "diamogli" is segmented into "diamo"→"give', "gli"→"him" (to him). Furthermore, it is also common to have two clitics combined (eg. "glie" + "le"). The problem with such clitic pronouns is that they are extremely frequent ending syllables in Italian. As for the "s" example in English, more rules were needed in order to prevent the segmenter from indiscriminately segmenting those syllables. For this reason, we implemented specific rules in order to segment them only when specific conditions are met.

The rest of the algorithm is similar to the English one. It segments suffixoids to then iteratively segmenting noun and adjective suffixes. Phonetical vowels are segmented as in the English algorithm.

Implementation notes

- The algorithms ignore punctuation. It is possible to remove the punctuation handler code if tokenised sentences are processed.

- The segmenters have a threshold of 3 characters beyond which they don't segment the words further. The result is something similar to using the NLTK Snowball stemmer R1, R2 or RV region.

- The ***while*** loop for noun/adjective suffixes can be replaced with a ***for*** loop and a fixed number of iteration. Although theoretically, words can have a large number of morphemes, they usually do not require more than one or two iterations of this loop.

- It is impossible to find a unique correct order for the various steps in the algorithms. Our approach is only one of the possible solution. Furthermore, we only implemented a limited number of special cases (eg. postverbal clitic pronouns in Italian). Further work is needed in order to perfect those algorithms.

3.1.1.2 Prefix Splitting

For prefix segmentation, we have developed a small script similar to Algorithm 1 which check whether a word starts with a prefix or not. Since boundaries between prefixes and roots of words are cleared compared to suffixes, we implemented a single language-independent algorithm. More specifically, there is one function that handles the segmentation process, however, the language needs to be specified in order to utilise the correct list[4] of prefixes. The list of prefixes is ordered in descending order based on the length of prefix (ie. longer prefixes are split before smaller ones). We set the segmentation number to 2 (no more than two prefixes are segmented). Moreover, we ignore single letter prefixes (eg. "a-" as in "acyclic") since the number of wrong segmentation is likely to drastically increase.

3.1.1.3 Oversegmentation and mergeBPE

As pointed out in the above sections, in order for a rule-based algorithm to produce a precise segmentation, specific rules need to be implemented for each affix. While there are rules that can be implemented quite easily like for the clitic pronouns above, it is not trivial for the majority of the affixes. For example, negative prefixes such as "un-", "in-", "ir-" or "im-" are widely used in English. However, it's not a simple task to disambiguate whether a word that starts with those letters contain the prefix or not, as we have discussed in Section 2.1.2.1.

Due to the fact that it is not feasible to implement rules for every single affix, our rule-based segmenter is affected by the oversegmentation problem, which means that words are segmented more than they should. For example, our segmenter consider the words "undo" and "union" in the same way, for this reason both "un-" are segmented. For this reason, we propose an unsupervised method to combine subwords back together, a modified version of BPE which we will call "*mergeBPE*".

As described in Section 2.3.3.1, BPE is an unsupervised segmentation technique that split a word into frequently seen subunits. At first, we thought that BPE would have started merging words when the number of merge operation exceeds the vocabulary size. However, for how it is implemented, BPE does not exceed the word bound-

[4]The lists of prefixes are obtained from the following resources:
English→https://en.wikipedia.org/wiki/English_prefix
Italian→https://en.wiktionary.org/wiki/Category:Italian_prefixes
German→https://en.wiktionary.org/wiki/Category:German_prefixes
Dutch→https://en.wiktionary.org/wiki/Category:Dutch_prefixes
Romanian→https://ro.wiktionary.org/wiki/Categorie:Prefixe_%C3%AEn_r_om%C3%A2n%C4%83

aries. We thus adapted BPE's source code to merge units together instead of splitting them into smaller units.

Learning mergeBPE

In order to get beyond the word boundaries, we changed perspective and applied BPE on a higher hierarchical level. The normal BPE implementation starts by individuating the words in the training data and merging character after character until the number of iterations is reached. Our mergeBPE does exactly the same, however, instead of words, we start from sentences (ie. entire sentences are put into the initial vocabulary) which are then split and merged.

Here is how we modified *learn_bpe.py*:

- *get_vocabulary()*: instead of adding the split words into the vocabulary, we save the entire sentence as an entry;
- *get_pair_statistics()*: for each vocabulary entry, we split it into words and count the frequency of the words pairs;
- *learn_bpe()*: we removed the line of codes that deal with the "end of word" symbol. We save in the model only the pairs that contain a prefix or a suffix. In order to do so, we replace the ***for*** loop with a ***while*** so that we can freely modify the merge operation counter. The reason for this is that we want to combine only the subwords within a word (so in this sense, mergeBPE doesn't exceed the word boundaries as for the classic BPE).

Applying mergeBPE

For *apply_bpe.py*, we replaced the *segment()* and *segment_tokens()* methods with a *merge()* method. Its purpose is to iteratively merge subwords until no more merge operations are allowed according to the BPE model learned above. The original BPE algorithm merges the units in accordance to their frequency (ie. more frequents pairs are merged together first). However, this approach might not be situated for our purpose.

For example, the Italian word "sperare" is wrongly segmented as "sp -er -ar -e" (correct segmentation "sper -are"). By merging the subunits pairs based on their frequency, the results would be: →"sp -er -are"→"sp -erare" (if "-erare" is a more frequent subunit than "sper").

Since the languages that we are going to analyse in this project (English, Italian, German, Dutch and Romanian) are defined by Dryer (2013) as "Strongly suffixing"[5],

[5]https://wals.info/feature/26A

we decided to opt for a left-to-right strategy in order to maximise the number of roots' subunit recombined together. Algorithm 4 shows the Python code for the left-to-right mergeBPE algorithm, where *suf_tok* and *pref_tok* are the suffix and prefix separators respectively.

Algorithm 4: Apply mergeBPE - left-to-right version

```
def merge(self, sentence):
    output = []
    word = sentence.split()
    if len(sentence) <= 1
        return ' '.join(word)
    for i in range(len(word)):
        if output:
            pair = output[-1] + word[i]
            if pair in self.bpe_codes_reverse.keys():
                output = output[:-1]
                output.append(pair)
            else:
                output.append(word[i])
        else:
            output.append(word[i])
        for i in range(len(output)): # Replace the special tokens
            pref = output[i].endswith(pref_tok)
            suf = output[i].startswith(suf_tok)
            new = output[i].replace(suf_tok, "").replace(pref_tok, "")
            output[i] = new + pref_tok if pref else new
            output[i] = suf_tok + output[i] if suf else output[i]
        return ' '.join(output)
```

Although mergeBPE might help mitigating the oversegmentation problem, it is not the ultimate solution. In fact there are two main issues with this approach:

- Frequent affix pairs are merged together: eg. "-ful" and "-ly" occurs frequently together so mergeBPE might learn to combine them when found. For example, in a word such as "beautifully", the suffix segmenter outputs "beaut -i -ful -ly" (which is correct), however, due to mergeBPE the final segmentation result is "beaut -i -fully";

- Balancing the number of mergeBPE operations: with the segmentation “sp -er -ar -e”, applying the left-to-right merge might result in the correct segmentation “sper -are”, however, with too many merge operations it can also result in “sperar -e”, or in the opposite way, if there are not enough merges, it will result in “sp -er -are” or not merging any subunits at all.

3.1.2 Unsupervised morphological segmentation

For this project, as unsupervised morphological segmentation, we have adopted Morfessor 2.0, a Python implementation of the Morfessor Baseline (Smit et al., 2014; Virpioja et al., 2013). Despite adding new features like semi-supervised learning or online training, the base components of this implementation remains the same as the original Morfessor baseline (Creutz and Lagus, 2005).

According to (Virpioja et al., 2013), Morfessor is composed of a lexicon, which stores the properties of the constructions, and a grammar, which actually define the constructions and how they combine with each other to create compounds. The main objective of Morfessor is to minimise the vocabulary size by discovering subwords that occur frequently across the words in a training set. To achieve this, it uses a generative probabilistic approach. The cost function is derived using *maximum a posteriori*, which means that given a training corpus D_W, the goal is to find the models parameters θ that maximise the following posterior probability:

$$\theta_{MAP} = \arg\max_{\theta} p(\theta|D_w) = \arg\max_{\theta} p(\theta)p(D_W|\theta)$$

While the actual loss function is to minimise the log probability:

$$L(\theta, D_W) = -\log p(\theta) - \log p(D_W|\theta)$$

To compute the likelihood $\log p(D_W|\theta)$ a hidden variable is introduced in order to avoid calculating the sum of the log probability of all the possible analyses (ways to segment the words).

3.1.3 Segmentation in practice

As in Sennrich et al. (2015) and Huck et al. (2017b), for all our morphological segmentation methods, we add special tokens for prefixes and suffixes, so that at postprocessing time, it is possible to merge the subunits to form complete words.

Original	Finally, the positive alternative meshes with our economic challenge and our national security challenge.
Morph (suffix+prefix)	Fin §§al §§ly , the posit §§ive altern §§at §§ive mesh §§e §§s with our eco## nom §§ic challeng §§e and our n §§a §§t §§ion §§al secur §§i §§ty challeng §§e .
Merge4K	Fin §§ally , the posit §§ive alternat §§ive mesh §§es with our econom §§ic challenge and our na §§tional secur §§ity challenge .
Merge16K	Final §§ly , the positive alternat §§ive mesh §§es with our econom §§ic challenge and our na §§tional securi §§ty challenge .
Merge64K	Finally , the positive alternat §§ive mesh §§es with our econom §§ic challenge and our na §§tional security challenge .
Merge128K	Finally , the positive alternative mesh §§es with our econom §§ic challenge and our na §§tional security challenge .
Morfessor	Final §§ly , the positive alter §§native mesh §§es with our economic challeng §§e and our national securit §§y challeng §§e .

Table 3.1: Example of various morphological segmentation strategies on a sentence from IWSLT'17 *dev2010*

We use "## "" after prefixes (eg. "un## "), " §§" before suffixes (eg. " §§ly"). As for Morfessor, since there is no distinction between prefixes and suffixes, we kept the suffixes notation " §§".

Table 3.1 shows an example of a sentence segmented with the different approaches described in the previous sections, training processes for these examples are described in Section 4.1.1.1. As discussed previously, the rule-based segmentation oversegment several roots ("final", "econom" and "nation") and it fails to segment correctly "positive". Our mergeBPE algorithm, with only a little number of mergeBPE, improve drastically the quality of the morphological segmentation, although with more mergeBPE operations more subwords reunite to form full words. As for Morfessor, it does a good job of segmenting relevant suffixes, although it misses some of them. A more detailed

analysis will be made in the following section.

3.2 Morphological Segmentation Task

In this section, we evaluate the above segmentation techniques on a morphological segmentation task to determine how morphological they actually are.

3.2.1 Method

We test the morphological level of the segmentation algorithms on the Gold Standard provided with the Morpho Challange 2010 (Kurimo et al., 2010). Since for this challenge only datasets for English, Finnish and Turkish were provided, we report scores for English only.

The English test set is composed of 1000 word/segmentation pairs (where multiple segmentations are allowed, we kept only the first one). Since no evaluation script is provided for the morphological segmentation task, we measure the performance of the segmenter with the following metrics:

$$precision = \frac{\#correct\_morphemes}{\#tot\_morphemes\_output} \tag{3.1}$$

$$recall = \frac{\#correct\_morphemes}{\#tot\_morphemes\_GS} \tag{3.2}$$

$$F_1 = 2 \cdot (\frac{precision \cdot recall}{precision + recall}) \tag{3.3}$$

where, for each word, $\#correct\_morphemes$ indicates the number of matching morphemes between the output segmentation and the Gold Standard, $\#tot\_morphemes\_output$ the total number of morpheme produced and $\#tot\_morphemes\_GS$ the total number of morphemes in the Gold Standard.

For this morphological segmentation task, we compare Morph (suffix+prefix splitter), Morph+MergeBPE (with {4k, 16k, 64k, 128k} merge operations) and Morfessor. In addition, we provide the results for BPE and BPE applied on top of the other morphological segmentation. The number of BPE merge operations is set to 32k.

MergeBPE, BPE and Morfessor are trained on the IWSLT'17 dataset described in Section 4.1.1.1.

Segmentation	Morph	4k	16K	64k	128k	M.ssor
Precision	41.03	40.85	41.60	44.01	46.40	**52.61**
Recall	50.64	41.19	40.12	39.81	42.03	**60.76**
F1-score	45.33	41.02	40.85	41.81	44.11	**56.40**

Table 3.2: Segmentation scores on the Morpho Challange 2010 segmentations' Gold Standard. xxK indicates Morph+Merge xxK

3.2.2 Performance

Table 3.2 show the results obtained by the different segmentation techniques on the morphological segmentation task.

Compared to the rule-based segmentation algorithm, Morfessor achieved the highest scores. By manually analysing the segmented test sets, we noticed that the main difference between Morfessor and Morph is compound splitting. In fact, while Morph is not designed to deal with compounds, Morfessor resulted to be extremely effective in finding compound segments (eg. "fire-power", "home-work", "god-parent"). As a drawback, it individuates and segments sequences of characters that are words but not compound segments (eg. "ingredient" $\rightarrow$ "ing-**red**-ient"). In addition, we found wrong segmentations that are not related to affixes or compounds (eg. "triumvirate" $\rightarrow$ "tri-um-vir-ate") which we attribute to the poor data quality of the training set.

As for Morph, we can see that it obtained a high recall but a low precision. This is because Morph is able to segment correctly most of the affixes but it fails in maintaining roots intact (eg. in an extremely morphological word such as "egotistically", Morph is able to find all the affixes "egot-ist-ic-al-ly", however, a word like "laboratory" is segmented in "lab-or-at-or-y"). When applying MergeBPE over Morph, there is an improvement in precision, however, the recall drops. This is because when applying MergeBPE, many affixes pairs are merged together (eg. *"ing+s", "al+ly", "ed+ness"*, ...), lowering the numerator in Eq. 3.1 and 3.2, while the denominator remains the same in the latter equation.

Similarly, we report the results obtained by BPE and the morphological segmentation with BPE applied on top of them in Table 3.3.

As expected, since BPE is not a morphological segmentation algorithm, it obtained the lowest scores while when applied to the morphological segmented words, it lowers both their precision and recall by $\approx$10 points. This loss is caused by the fact that the subwords are segmented even more. As a consequence, the total number of subwords

Segmentation	BPE	Morph	4k	16K	64k	128k	M.ssor
Precision	16.79	32.59	30.72	28.90	27.44	27.17	**36.80**
Recall	19.88	50.60	41.10	39.05	37.55	37.19	**51.79**
F1-score	18.21	39.65	35.16	33.22	31.71	31.40	**43.03**

Table 3.3: Segmentation scores for morphological segmentation+BPE on the Morpho Challange 2010 segmentations' Gold Standard. xxK indicates Morph+Merge xxK

Segmentation	BPE	Morph	4k	16K	64k	128k	M.ssor
w/o BPE	**2253**	2849	2339	2240	2105	2030	2602
w/ BPE	2668	3498	3014	3045	3083	3030	3171
Difference	-	+649	+675	+805	+978	+1000	+569

Table 3.4: Number of subwords in the MorphoChallenge 2010 test set segmented with the various segmentation techniques with (w/) or without (w/o) BPE. In the first cell(BPE w/o BPE) the number of morphemes in the Gold Standard.

increases (as we can see from Table 3.4) while the number of correct morphemes decreases. We should also note that, in contrast with the results in Table 3.2, by increasing the number of mergeBPE merge operations, the precision of the models decrease. We can explain this phenomenon by considering the effect of applying the two algorithms. In fact, we used mergeBPE in order to rebuild segmented roots while BPE splits rare words/subwords into smaller units. As we have noticed above, one of the consequences of using mergeBPE is that frequent affix pairs are merged together. For this reason, when combining mergeBPE and BPE we have that the number of correct morphemes segmented is lowered even more. For instance, "egot-ist-ic-al-ly" with 64K mergeBPE operations becomes "egot-istic-ally" reducing the number of correct morphemes from 5 to 1, but by applying BPE on top of it the result is "e-got-istic-ally" with 0 correct morphemes. Although the scores for all the models are reduced, the loss is greater for models with higher mergeBPE operations since more subwords are split by BPE as we can see from Table 3.4.

Chapter 4

Translation Experiments

In this section, we aim to answer empirically whether a morphologically-driven segmentation technique improves the performance of an NMT system. In the first part, we describe the setup of our experiments and the preprocessing process, while in the second part, we discuss the performance of the BPE and morphological-based models in different training conditions and settings.

4.1 Experiments setup

4.1.1 Dataset

We perform our experiments on the multilingual parallel corpora provided for the IWSLT'17 shared task (Mauro et al., 2017) (a more complete description of the dataset is provided in the original paper). The dataset is composed of five languages (English, Italian, German, Dutch and Romanian) with a total of twenty translation directions with approximately 200K sentences per direction. For the Zero-Shot experiments, four directions are retained from the training set (Italian↔Romanian and German↔Dutch).

4.1.1.1 Preprocessing

For all the experiments the data is tokenised and truecased with the scripts provided in the Moses toolkit (Koehn et al., 2007). In addition, for the training set, sentences longer than 80 units are excluded.

BPE is jointly trained on all the training files with the amount of BPE operations set to 32K. The resulting dictionary contains units for all the five languages and it is shared by both the encoder and the decoder.

The parallel corpora are used as monolingual corpus for training Morfessor. Each language is trained separately on around 800K sentences, however, since a number of TED talks are shared, the number of unique sentences should be considerably lower (probably around 300K).

We also perform experiments with the rule-based morphological segmentation algorithms and the modified version of the BPE described in Section 3. The number of mergeBPE operations was set to 0, 4K, 16K, 64K and 128K. We will refer the *suffix + prefix* rule-based splitter as *Morph* and *MergexxK* the merge BPE algorithm[1], where *xx* denotes the number of merge operations.

For all the models, the dataset is preprocessed in the following order:

1. Remove all the XLM tags and meta-data;
2. Append the target token at the beginning of each source sentence;

   Optional 1a: (Morph) Apply the rule-based morphological segmentation;

   Optional 2a: (mergeBPE) Learn and apply the modified BPE for merging words;
3. Tokenise the sentences;

   Optional 1b: (Morfessor) Learn and apply Morfessor;
4. Learn and apply a truecaser;
5. Learn and apply BPE;
6. Merge the training files to create a single training corpus (and similarly for the dev set)

Optional 1a and 2a are applied to all the Morph-based models (except for the Morph model without mergeBPE version which includes only Optional 1a). Optional 1b is applied to the Morfessor based only. The BPE baseline does not include any of the above optional steps.

Statistics of the source corpus are shown in Table 4.1. Since both source and target corpora are preprocessed as described above, target corpus' statistics should be similar to the source ones. The first half of the Table shows statistics for tokenised data (before step 4), while show statistics on data where BPE is applied (step 5). Morph correspond to Optional 1a, merge xxK to Optional 2a and Morfessor to Optional 1b.

[1]Since mergeBPE is applied only after Morph, we will use *merge xxK* to refer to Morph+mergeBPE as well.

Preprocessing	#types	#tokens
Tokenised	430 K	85 M
Morph	219 K	126 M
Morph + merge4k	222 K	107 M
Morph + merge16k	228 K	101 M
Morph + merge64k	252 K	96 M
Morph + merge128k	279 K	95 M
Morfessor	98 K	115 M
BPE	32 K	98 M
Morph + BPE	32 K	131 M
Morph + merge4k + BPE	32 K	112 M
Morph + merge16k + BPE	32 K	108 M
Morph + merge64k + BPE	32 K	107 M
Morph + merge128k + BPE	32 K	106 M
Morfessor + BPE	31 K	117 M

Table 4.1: Source sentences statistics

4.1.2 Settings

The evaluation of the effect of different segmentation techniques is empirically evaluated using the Nematus toolkit (Sennrich et al., 2017b), which is an implementation of an Encoder-Decoder RNN with attention mechanism and GRU cells.

Unless differently specified, we use a shallow architecture (one bi-directional layer for the Encoder and one layer for the decoder) with word embeddings of size 512 and hidden layers of size 1024. We train our models with minibatches of size 80, maximum sentence length of 50, dropout with probability 0.1 on full words and with probability 0.2 on all the other layers as in Sennrich et al. (2016) and layer normalisation as in Sennrich et al. (2017a).

We train the models with Adam optimiser (Kingma and Ba, 2014) and a learning rate of 0.0001. We validate the models every 10 000 updates with the provided dev set (*dev2010*) and do early stopping when the validation cost (cross entropy) has not decreased in 10 updates.

It's important to note that since the objective of the project is not to obtain state-of-the-art results but rather to analyse the effect of different segmentation strategies on NMT systems, the above hyperparameters are not fine-tuned.

L1→L2	de	en	it	nl	ro
de	-	31.24 29.95	19.09 18.52	22.72 21.01	16.92 16.25
en	26.53 24.5	-	27.65 25.89	29.88 29.00	25.23 24.62
it	19.54 18.24	31.54 30.55	-	20.65 19.70	19.42 18.54
nl	23.28 22.21	35.54 34.12	20.49 19.78	-	18.54 17.83
ro	20.63 19.89	33.26 31.72	21.68 21.02	20.98 20.10	-

(a) Multilingual

L1→L2	de	en	it	nl	ro
de	-	31.41 29.92	19.22 17.61	20.26 17.89	17.06 16.47
en	26.80 24.96	-	27.67 25.29	29.99 28.77	25.62 24.2
it	19.05 17.57	31.74 29.28	-	20.51 19.00	16.42 15.04
nl	20.45 19.02	35.62 33.78	20.63 18.59	-	18.76 17.73
ro	20.27 19.11	32.96 31.52	18.10 15.73	20.61 19.81	-

(b) Zero-Shot

Table 4.2: Results for the *tst2010* set. In each cell, the top score is for the BPE-based model while the bottom one is for the Morph+BPE one (without mergeBPE)

Performance of the systems are measured with BLEU (Papineni et al., 2002) and chrF3 (Popović, 2015). To compute BLEU scores we use *multi-bleu-detok.perl* which uses the same internal tokenisation of *mteval-v13a.pl* but takes plain text as input instead of XML files, while for chrF3 we use the implementation of the original author[2].

4.2 Experiments results

For readability reasons, only the relevant results are shown in this Section. However, the scores for all models' directions can be found externally[3] as described in Appendix A.

4.2.1 BPE vs Morphologically-Driven Segmentation

We first provide a comparison between models trained on parallel corpora segmented with BPE and Morph on the multilingual and Zero-Shot tasks. The results for these models are shown in Table 4.2.

On the contrary of what we were expecting, our morphological segmentation not only did not improve the translation quality compared to BPE, but can be even considered harmful. In fact, on average the BLEU score dropped by -1.07 points for the multilingual model and by -1.60 for the Zero-Shot one (BPE and Morph column in

[2]https://github.com/m-popovic/chrF

[3]For this project 27 models were trained, each evaluated on 20 translation directions, 2 test sets and 2 evaluation metrics. A number of results impossible to fit in a reasonable space.

Segmentation		BPE	M.ssor	Morph	4k	16K	64k	128k
Multi	*tst2010*	**24.24**	23.49	23.17	23.53	23.85	24.08	24.10
	tst2017	**21.88**	20.64	20.36	21.04	21.05	21.37	21.45
Zero	*tst2010*	**23.66**	22.95	22.06	23.13	23.47	23.26	23.22
	tst2017	**17.89**	17.18	16.83	17.49	17.52	17.54	17.37

Table 4.3: Average BLEU scores on test sets for multilingual and Zero-Shot tasks with different segmentation strategies. Morph+MergeBPE is abbreviated with the number of merge operations.

Table 4.3). This result is in sharp contrast with our initial hypotheses for which we believed that an NMT would have benefited from a more morphologically-informed segmentation and that the effect would have been greater in Zero-Shot conditions.

We think that the reasons for these unexpected results can be found in the balance between the benefits of morphological segmentation and the loss derived from a wrong segmentation. As described in Section 3.1.1.3, the main problem of our morphological segmentation technique is oversegmentation for which a great number of word roots are split into smaller units. This leads to two consequences:

- It is harder to learn the affixes' embeddings correctly because the associated subunits are used to represent both the affixes themselves and a number of roots' parts. This affects more affixes which appear frequently at the beginning/end of roots because the model will have to disambiguate them over a broad range of meanings;

- It is harder to learn the meaning of the roots since the model has to reconstruct them from unrelated units (unless there are enough training examples).

This theory is supported by the performance of the morphological-driven models for which MergeBPE has been applied, as shown in Table 4.3.

As we can see from these results, by applying MergeBPE, the average performance of the morphologically-driven model improved considerably, especially for the Zero-Shot task. With only a few number of merge operations such as 4K, on *tst2010*, the average BLEU scores increased by +1.07 for the Zero-Shot model but only by +0.36 for the multilingual one. Another interesting fact is that, while for the multilingual task the average BLEU score kept increasing with the number of merge operations, for the Zero-Shot, the best performance is reached with 16K/64K MergeBPE operations.

Direction	*de→nl*	*nl→de*	*it→ro*	*ro→it*
BPE	20.26	**20.45**	**16.42**	**18.1**
M.ssor	19.55	20.16	14.97	16.82
Morph	17.89	19.02	15.04	15.73
4k	19.48	19.89	16.32	17.39
16K	20.12	20.2	16.28	16.37
64k	**20.52**	20.33	15.94	17.13
128k	20.28	20.16	16.21	16.87

Table 4.4: BLEU scores for Zero-Shot directions on *tst2010*

Despite these improvements, none of our morphologically-driven models were able to outperform BPE[4].

A further reason that can explain the gap between the BPE and Morph-based model is proposed by Huck et al. (2017b). Although all their morphological models obtained a better score over their baseline, their best model was the suffix+BPE model. In fact, they noticed that by preprocessing the data with the prefix splitter, the performance of the model deteriorates. They attribute this behaviour to the fact that prefixes change drastically the meaning of a word, for this reason, when prefix segmentation is applied, the embedding layers become less effective since the stem may be confused with a completely different word. This can also explain the drastic improvements to Morph when mergeBPE is applied. In fact, as described in Section 3.1.1.3, the algorithm merges subwords from left to right, as a consequence prefix segmentation is reversed first.

In addition to the rule-based segmentations, Table 4.3 shows the performance of the model trained with Morfessor segmented data. Despite achieving the highest score for the morphological segmentation task, it wasn't able to beat any MergeBPE model, while outperforming only the base rule-based segmentation technique[5].

Table 4.4 shows the BLEU scores of the various models in translating the Zero-Shot directions in *tst2010*[6]. The overall pattern is consistent with the average BLEU scores in Table 4.3. Looking more closely, we can see that our Morph+merge64K had

[4]chrF3 scores externally as describe in Appendix A. They are not reported in this Section because they are consistent to BPE and do not provide any additional insight.

[5]A more detailed discussion of the relation between translation and morphological segmentation is provided in Section 5.1.

[6]Results on *tst2017* are consistent with the ones shown in this Table 4.4.

Direction	*de→it*	*en→it*	*nl→it*	*ro→it*
BPE	**19.22**	**27.67**	**20.63**	**18.1**
M.ssor	18.3	26.67	19.7	16.82
AVG Morph	18.61	26.93	19.79	16.7

Table 4.5: BLEU scores for directions to Italian on *tst2010* (Zero-Shot models). AVG Morph is the average BLEU scores of Morph+$\{0,\cdots,128K\}$ MergeBPE models.

an improvement over BPE of +0.26 points for *de→nl*, while on the other hand, all our morphologically-informed models performed considerably worse with respect to BPE, especially *ro→it* for which there is a ≈-1.4 difference in BLEU scores.

From Table 4.4 we might also notice that, in contrast to the average performance of the models, Morfessor achieved a higher score on *ro→it*. One of the possible causes is the difference in segmentation algorithm used for the Italian suffixes since it was implemented from scratch and not based on the NLTK stemming algorithm. In fact, with respect to the NLTK algorithm, our segmenter handles almost twice the number of suffixes. Thus, as a counter-effect, a more problematic oversegmentation. In Table 4.5 we can see that this phenomenon happens only for the Zero-Shot direction. For this reason, we can claim that Zero-Shot translation benefits from morphological information to a certain degree, but at the same time, it is more sensitive to the problems derived from oversegmentation.

4.2.2 Low-Resource

In this section, we investigate how reducing the training dataset size affects the performance of BPE and a morphologically-driven segmentation method. For these experiments, we train our model with 10%, 30%, 50% and 70% of each translation direction.

Table 4.6 summarises the average performance of BPE and Merge64K, while Table 4.7 and 4.8 take a closer look at the models' scores for the Zero-Shot directions.

While on average BPE consistently outperforms the morphological-driven model in a similar way to the results in Section 4.2.1, there is no clear pattern for the Zero-Shot directions.

On the lowest resource condition, the two models achieved comparable BLEU score with BPE which still performed slightly better on all the directions. The gap between the Zero-Shot translations of the two models increases with the size of the training data, on average, from -0.33 to -1.2 as shown in Figure 4.1. The only out-

%	Model	Average BLEU		Average chrF3	
		tst2010	*tst2017*	*tst2010*	*tst2017*
10	Morph	13.60	10.66	35.62	32.49
	BPE	**13.77**	**10.8**	**35.67**	**32.54**
30	Morph	18.8	15.65	42.13	39.16
	BPE	**19.26**	**16.18**	**42.39**	**39.67**
50	Morph	20.43	17.72	44.39	41.45
	BPE	**21.14**	**18.03**	**44.44**	**41.65**
70	Morph	21.95	18.88	45.46	42.77
	BPE	**22.55**	**19.48**	**46.06**	**43.36**

Table 4.6: Average scores of all translation directions for low-resource (Zero-Shot) models (BPE vs Morph+64K)

lier appears to be the medium size model (50%) for which the morphological model outperformed BPE for *de→nl* and *it→ro* on both test sets and *nl→ro* on *tst2010*.

Table 4.8 shows the chrF3 scores for the above experiments. The results are overall consistent with BLEU, however, there is a little number of cases where the two metrics don't match. The differences don't appear to be significant and they can be attributed to precision and recall bias of the two metrics as in Sennrich et al. (2015).

Following the reasoning of the previous section, a possible explanation of these results may lay on the models' dictionaries and the different learning problems that affect the two methods. On one hand, the morphological model has a dictionary composed by affixes and roots/segmented roots (in addition to other subwords). In this case, the model would benefit from the morphological segmentation only if it has enough data to learn the patterns in the languages and to learn to reconstruct words from segmented roots. On the other hand, BPE generates a vocabulary containing frequent full words and subwords, so the model can easily learn a good representation for those full words (because they are frequent and thus have a lot of examples) but it has to learn inconsistent an pattern between the subwords.

In a low resource condition, we believe that the morphological model was not able to learn a good representation for the affixes (also considering that there are many rare and ambiguous affixes) nor for the frequents words because they are wrongly segmented. On the contrary, while BPE-based model performance can be comparable to the morphological one, as suggested above, it can achieve a slightly higher BLEU

%	Model	de→nl		nl→de		it→ro		ro→it	
		tst2010	*tst2017*	*tst2010*	*tst2017*	*tst2010*	*tst2017*	*tst2010*	*tst2017*
10	Morph	12.07	8.62	11.74	7.73	8.53	6.86	9.17	8.05
	BPE	**12.4**	**8.63**	**12.45**	**8.56**	**8.61**	**6.87**	**9.57**	**8.28**
30	Morph	15.51	12.12	16.11	11.96	12.2	**11.09**	**13.10**	11.74
	BPE	**16.93**	**13.29**	**17.08**	**13.18**	**12.65**	10.91	12.74	**11.75**
50	Morph	**17.67**	**13.95**	**17.97**	13.61	**13.50**	**12.38**	14.58	13.31
	BPE	16.82	12.94	17.84	**14.12**	12.89	12.31	**15.07**	**14.95**
70	Morph	18.84	**15.40**	19.48	14.95	13.48	13.29	14.00	13.25
	BPE	**18.95**	14.93	**19.57**	**15.52**	**16.19**	**13.79**	**16.75**	**15.46**

Table 4.7: BLEU scores on Zero-Shot directions with different resource levels (BPE vs Morph+64K)

score by translating correctly the unsegmented frequent words. Continuing on this line, we would expect that with more training sentences, the morphological model will eventually outperform BPE as a better segmentation will likely result in a better representation of the language. This claim is also supported by the results achieved by Huck et al. (2017b) where their morphological model obtained a better score compared to BPE when trained with 1.7M sentences. We should point out that the affixes themselves are not learned by the model since the rule-based segmentation algorithm split the affixes according to a predefined list of affixes. However, representations of these affixes need to be learned. For this reason, with low-resource conditions, a morphological segmenter might not be as effective as we hypothesised.

Regarding the results of the 50% dataset model, there are two possible ways to explain the morphological model's positive results. The first one is that the morphological model was able to learn a good representation of frequent words in addition to some morphological patterns (e.g. frequent inflections) for which BPE fails at translating because of the non-morphologically-driven segmentation. The second one can be attributed to the inconsistency of Neural Networks, in fact, it is possible that the training process stopped on an unexpectedly good minimum (or the BPE-based model terminated on a bad one), so a repetition of the experiments described in this project may lead to a slightly different result.

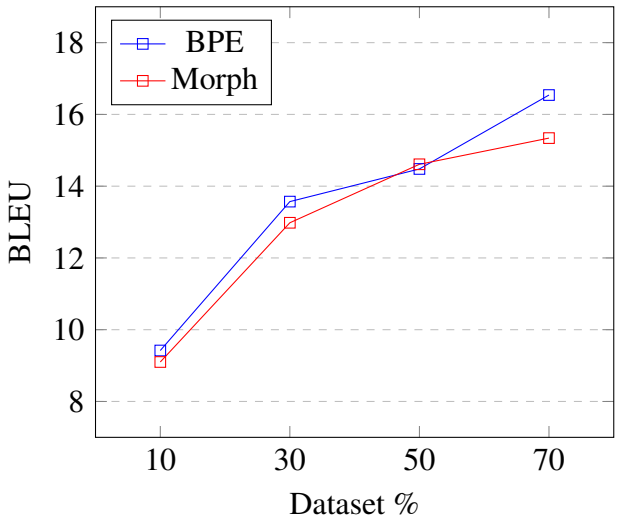


Figure 4.1: Average BLEU scores on Zero-Shot directions (both *tst2010* and *tst2017*) for BPE and Morph+64K

4.2.3 More settings

4.2.3.1 Making things bigger

The following experiments will aim to validate the results of the previous sections. More specifically we try to verify whether our experiments are indeed architecture-independent. We report the results obtained by two architectural variations:

- Deep model: 4 bi-directional Encoder layers and 8 Decoder layers;
- Larger embeddings: word embedding size of 1024 instead of 512 (the hidden layer size remains 1024).

Table 4.9 shows the average BLEU and chrF3 of the Zero-Shot models trained with different architectures. Both BPE and Morph64K improved their performance by increasing the depth of the architecture. However, this variation benefits the two model equally so Morph64K did not outperform BPE. Similarly, in the models with larger word embeddings the gap between the two models remained the same but in contrast to the deep models, their scores are lower than the original ones. Overall, it seems that the results obtained in the previous experiments are not related to the architecture of the models. The performance of the various models are consistent with the previous findings.

%	Model	de→nl		nl→de		it→ro		ro→it	
		tst2010	*tst2017*	*tst2010*	*tst2017*	*tst2010*	*tst2017*	*tst2010*	*tst2017*
10	Morph	33.41	29.45	33.27	29.95	28.95	27.19	30.9	**29.29**
	BPE	**33.42**	**29.96**	**33.95**	**30.27**	**29.51**	**28.03**	**31.13**	29.25
30	Morph	37.66	34.37	38.93	35.36	34.91	33.86	**37.13**	**35.34**
	BPE	**39.89**	**36.47**	**40.2**	**36.56**	**35.55**	**34.38**	35.52	34.43
50	Morph	**40.54**	37.36	**41.41**	37.20	**36.13**	**34.86**	38.06	36.56
	BPE	39.16	**37.59**	41.21	**37.39**	35.06	34.74	**38.25**	**37.61**
70	Morph	**42.69**	**39.39**	42.97	39.02	35.98	35.95	37.16	36.10
	BPE	42.20	38.69	**43.02**	**39.45**	**39.57**	**38.44**	**40.87**	**39.35**

Table 4.8: chrF3 scores on Zero-Shot directions with different resource levels (BPE vs Morph+64K)

Model		Average BLEU		Average chrF3	
		tst2010	*tst2017*	*tst2010*	*tst2017*
Deep	BPE	**25.01**	**22.44**	**48.34**	**46.28**
	Morph64K	24.50	21.81	47.79	45.46
Large Emb.	BPE	**23.13**	**20.87**	**46.62**	**44.72**
	Morph64K	22.79	20.19	46.30	44.01
Shallow (base)	BPE	**23.66**	**21.20**	**47.11**	**45.81**
	Morph64K	23.26	20.64	46.81	44.59
	bpe2morph64K	23.10	20.53	46.62	44.48

Table 4.9: Average scores for BPE and Morph64K with different architectures.

4.2.3.2 Morphological Target-Side

So far, all the models were based on the same segmentation strategy on both source and target sides (i.e. bpe2bpe, morph2morph, morfessor2morfessor), however, Huck et al. (2017b) reported an improvement in translation quality when injecting morphological information only on thetarget-side[7] in comparison to a BPE baseline. However, Huck et al. (2017b) did not provide a motivation for the only-target-side morphological model, other studies for which morphology was applied only on the target-side such as Tamchyna et al. (2017) and Passban et al. (2018) attribute this decision to Belinkov et al. (2017)'s findings. In fact, according to their studies, the NMT Decoder is not

[7]Huck et al. (2017b) do not report any experiments of morphological models on both source and target side.

capable of capturing morphological information in contrast to the Encoder. For this reason, injecting morphological information into the Decoder is a sensible choice.

In this experiments, we compare the performance of the BPE-based and Morph64K-based models to one trained with BPE on source side and with Morph64K on target side (bpe2morph64k). The preprocessing is similar to the one described in Section 4.1.1.1, however, the optional steps 1a and 2a are applied only on target side, while the final BPE is trained separately for source and target which results in two distinct dictionaries.

The results for bpe2morph64K are shown in Table 4.9. As we can see from the average scores, the morphological target-side models not only wasn't able to outperform the BPE baseline but did worse than the fully morphological model as well. We believe that this result is not to be attributed to the difference in segmentation algorithm between source and target but rather on how BPE was trained. In fact, bpe2morph64's BPE was trained separately for source and target, while for the other models it was trained jointly. Although it is not always the case that an NMT model benefits from joint BPE, it might be possible that multilingual/Zero-Shot models benefit more from joint BPE since source and target languages are the same.

Chapter 5

Analysis

In this section, we take a closer look at the relation between morphology and translation. More specifically, we first discuss the correlation between the results obtained in the morphological segmentation and translation tasks described in Section 3 and Section 4, to then make an in-depth analysis of the areas where morphological segmentation affect positively or negatively translation quality.

5.1 Translation and Morphological Segmentation

In Section 3.2 we found that the most morphological segmentation technique was Morfessor. Morph was able to identify most of the affixes but it failed in maintaining roots intact leading to a low precision score. This problem is slightly overcome by mergeBPE, however, we noticed that in addition to reconstructing some of the roots, it combines together frequent affixes pairs as well, affecting negatively the scores in the segmentation task.

In contrast, in Section 4.2.1, the Morfessor-based model wasn't able to beat any of the MergeBPE-based ones in the translation experiments, outperforming only the Morph-based model.

Although a direct comparison between translation and morphological segmentation performance would result in an inverse correlation between the two tasks, we believe that there are more factors that influence their relation, most of all, the information level of the subwords.

As we have seen from the results reported so far on the translation task, a different segmentation affects drastically the performance of the model. Although the size of the vocabulary remains approximately the same and the dataset does not change, different

segmentations for the same word may convey different information. For instance, in Section 3.2 we have demonstrated that roots reconstructed by mergeBPE are split again by BPE. This phenomenon occurs for many subwords in the data, however, a root segmented with BPE results in a better translation compared to a wrong morphological segmentation. Moreover, the number of correct segmented morphemes does not impact the translation quality directly. In fact, maintaining roots intact seems to be more valuable than maintaining affix pairs separated. As we have discussed in Section 4, having enough data for a subword makes all these distinctions pointless.

To sum up, morphological segmentation accuracy is not directly proportional to translation quality since a different "wrong" morphological segmentation can lead to different degrees of translation quality.

5.2 Translation Quality with Contrastive Pairs

In order to have a better insight into the actual translation quality of the various systems, we first test them on the Contrastive evaluation task proposed by Sennrich (2016). The objective of this evaluation is to automatically assess the grammatical quality of an MT model by analysing its ability in recognising a correct translation instead of an ungrammatical one.

5.2.1 Method

The test set is composed of ≈ 97000 $En \rightarrow De$ contrastive translation pairs. Each pair is formed of a correct reference translation and a sentence in which one of the following grammatical errors is introduced:

- *noun phrase agreement*: noun's number or gender is changed to disagree with its determiner;
- *subject-verb agreement*: the number of a verb is modified to introduce disagreement between subject and verb;
- *subject adequacy*: the number of a verb is modified to change the meaning of the sentence;
- *auxiliaries*: "have" and "be" auxiliaries are swapped;

Error	BPE	Merge64K	Morfessor
noun phrase agreement	83.44	**87.60**	86.57
subject-verb agreement	84.50	**86.80**	84.54
subject adequacy	80.91	**84.60**	84.12
auxiliary	82.91	**86.28**	84.26
verb particle	73.96	83.37	**85.47**
polarity	87.58	**93.61**	93.08
compound	72.20	73.64	**78.70**
transliteration	**90.01**	89.74	86.33

Table 5.1: Accuracy (in percent) of BPE and morphological-based models on contrastive errors.

- *separable verb particle*: German separable prefixes are replaced by one unseen in the WMT'16 training data (Bojar et al., 2016);
- *polarity*: polarity of the sentence is reversed by adding or removing negative/-positive particles, determiners or prefixes;
- *compound*: swap the segments in a compound;
- *transliteration*: two adjacent characters are swapped in proper nouns that have to be transliterated.

For each error type, we compute the accuracy of the NMT system in assigning a higher score to the correct reference translation. We compare the accuracy on contrastive pairs for the BPE, Morph64k and Morfessor-based models described in Section 4.

5.2.2 Results

As we can see from the results for the contrastive pairs evaluation shown in Table 5.1, both Morfessor and Merge64K achieved a considerably higher accuracy on almost all the contrastive error categories compared to BPE.

We summarise the results as follows:

- *Affixation*: on all the categories related to suffixes or prefixes, the morphological models outperform BPE, with Merge64K obtaining the highest scores. This

can be explained by the fact that both Morph and Morfessor are able to identify the affixes involved in the various contrastive pairs (eg. affixes for gender or number), however, since Morfessor is an unsupervised segmenter, it is not able to split them in a rigorous way, in contrast to Morph which is rule-based. Furthermore, a better affix segmentation seems to make the model more robust on longer dependencies as shown in Figure 5.1. In fact, while the gap between Merge64K and BPE remained constant, Morfessor's performance is lower for agreement between distant words.

- *Compound*: Morph+MergeBPE does not implement any compound recogniser. On the other hand, as we have discussed in Section 3.2, Morfessor is extremely good at finding compound segments in words. As a result, Merge64K did not do much better than BPE, while Morfessor was able to make 6.5% fewer errors.

- *Transliteration*: this is the only category where BPE made fewer errors. BPE is well known to perform extremely well on such problem as shown in Sennrich et al. (2015) and Sennrich (2016). A possible reason why Merge64K and Morfessor make more transliteration error is that they individuate affixes in proper nouns. For instance, "Peter" will be split in "Pet er" with "er" not being just a subword but a suffix that will be translated instead of transcribed.

We conclude from the above results that there is a strong correlation between segmentation and grammaticality and that with a linguistic-informed segmentation, the translation models are able to capture better words' internal structure and connections with the surrounding words.

5.3 Manual Analysis

In this section, we manually analyse sentences from *tst2010* and *tst2017* in order to find the areas affected positively or negatively by our morphological segmentation. More specifically, the purpose of this manual analysis is to validate and discuss the results obtained in Section 5.2 with sentences from different translation directions. In order to shrink the number of sentences to analyse, we decided to consider only sentences containing unseen or rare[1] words. For an easier evaluation, we will consider only

[1]We define *rare* words as those with less than four occurrences in the training set. As we have mentioned in Section 4.1.1.1, the same TED talk may be repeated up to four times (eight if we consider both the source and target languages). For this reason, with a lower "threshold", words that occurred multiple times in the dataset but in only one talk will not be considered.

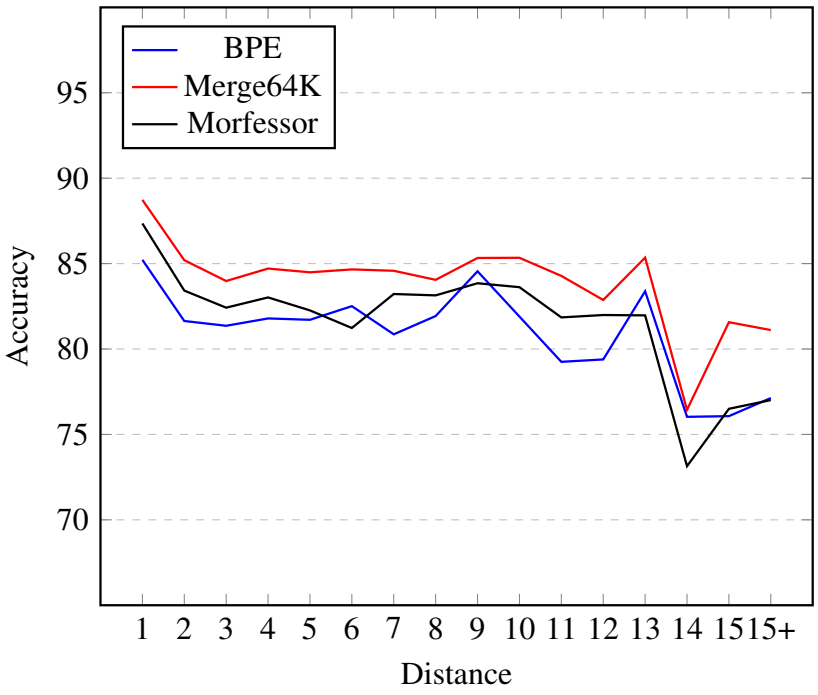


Figure 5.1: Subject-Verb agreement accuracy in relation to the relation distance between subject and verb.

translations between English and Italian.

5.3.1 Transliteration Problem

As resulted from the contrastive evaluation described in Section 5.2, BPE makes fewer transliteration errors compared to our morphological model. By manually analysing the translations from *tst2010*, we notice that BPE was able to transliterate correctly all the analysed examples in the test set while Morph64K only around half of them. On the evaluated examples we found different scenarios, however, we can group them into two different categories:

1. *Incorrect transliteration*: the majority of the cases where Merge64K fails in transliterating a word is because of wrong morphological segmentation. As we can see in Table 5.2, since the segmentation algorithm is rule-based, many proper nouns are segmented with affixes' subunits (eg. "-o", "-oto", "-al", etc...). While subwords that end with the BPE token "@@" are correctly transliterated, wrongly segmented affixes are translated to affixes in the target language. It is also worth to mention that we did not find any correct affix translation either when dealing with proper nouns (eg. the Italian suffix "§§o" should be translated

Word	BPE		Morph64K	
	input	output	input	output
Thabo	T@@ hab@@ o	T@@ hab@@ o	T@@ hab §§o	T@@ h §§o
Erodoto	E@@ ro@@ d@@ oto	E@@ ro@@ d@@ oto	Er@@ od §§oto	Er@@ od §§y
Kean	Ke@@ an	Ke@@ an	K §§e@@ an	K@@ ed §§ian
Bhopal	B@@ hop @@al	B@@ hop §§al	B@@ hop §§al	B@@ hop@@ al
Offit	Off@@ it	Off@@ it	Off@@ it	Off@@ it
Azeroth	A@@ zer@@ oth	A@@ zer@@ oth	A@@ zer@@ oth	A@@ zer@@ oth

Table 5.2: Transliteration with different segmentations (Italian on source and English on target). First half are correct transliteration for BPE only, second half for both BPE and Merge64K.

to a null morpheme but, instead, it is transliterated).

2. *Correct transliteration*: Merge64K was able to transliterate proper nouns in almost half of the cases. However, by looking at the segmentation of those words we notice that in almost all of these cases it got a correct transliteration because the segmentation of the words was the same for both BPE and Merge64K.

In addition to the above, we found a small number of cases where BPE and Merge64K had the same segmentation but with different subunits (eg. the first and fourth example in Table 5.2). However, Merge64K's performance on those words wasn't consistent, in fact, there were both correct and incorrect transliterations.

Overall, BPE resulted to be highly robust with transliteration while Morph64K was able to transliterate a word only if it had the same segmentation as BPE.

A possible solution to solve the incorrect transliteration problem is to use a single separator token instead of multiple to differentiate suffixes, prefixes and BPE, similar to the approach used by Banerjee and Bhattacharyya (2018). As we have seen from the examples in Table 5.2, the main problem in transliterating is the ambiguity brought by the affixes token. In addition to helping transliteration, using a single token may help to mitigate ambiguity caused by oversegmentation as well since the two problems seem to have the same root cause.

5.3.2 Translating tenses

Verbs are known to be problematic in Machine Translation, especially when translating to a morphologically-richer language. For example, compared to English, romance

System	Sentence
Src	**Would you choose** the same vacation? And if you **would choose** a different vacation [...]
Ref	**Scegliereste** la stessa vacanza? E se **poteste scegliere** una vacanza differente [...]
BPE	**Scegliete** la stessa vacanza? E se **scegliete** una vacanza diversa [...]
M.64K	**Sceglieresti** la stessa vacanza? E se **sceglieste** una vacanza diversa [...]
Src	She **wasn't going to vaccinate** her kid against polio
Ref	Lei **non avrebbe vaccinato** suo figlio contro la polio
BPE	**Non vaccinava** il figlio contro la poliomielite.
M.64K	**Non avrebbe vaccinato** il suo bambino contro la poliomielite.
Src	[...] the fish are eating what they'**d be eating** in the wild.
Ref	[...] i pesci mangiano esattamente ciò **che mangerebbero** in un ambiente selvatico.
BPE	[...] i pesci mangiano quello **che mangiavano** in natura.
M.64K	[...] i pesci mangiano quello **che mangerebbero** in natura.
Src	**I'm only showing** the top 500 most popular Wikipedia pages right here.
Ref	**Vi mostro** solo le prime 500 tra le più visitate su Wikipedia.
BPE	**Vi mostro** solo le 500 pagine più popolari di Wikipedia qui.
M.64K	**Sto mostrando** solo le 500 pagine di Wikipedia più popolari proprio qui.

Table 5.3: Verb translations for BPE and Merge64K. "[...]" indicates that the sentence shown is only a segment of a longer sentence.

languages such as Italian are much more complex in terms of verb tenses. Italian has 21 verb tenses while English only has 12. In addition, half of those tenses need to agree with the gender of the subject.

In this section, we manually analysed the ability of BPE and Merge64K to translate tenses from English to Italian. First of all, for the vast majority of the sentences, there were not any divergences between the two models' translations. This is because, since the dataset is based on spoken language, most of the verbs fall into a few common tenses. However, we noticed that Morph64K, compared to BPE, is more effective in translating into rare or complex tenses such as conditionals and subjunctives. Table 5.3 shows some of these translations.

While Merge64K tries to use more specific tenses. BPE seems to prefer using simpler ones such as the Italian equivalent of Simple Present or Simple Past. Despite the fact that such translations cannot be considered fully grammatically incorrect, a wrong conjugation results in a less fluent sentence and in a different meaning respect to the source. For instance, "*non vaccinava il figlio*" means *"she didn't vaccinate her kid (herself)"*, or *"quello che mangiavano in natura"* means *"what they ate in nature"*.

Lastly, we found several translations provided by Merge64K that are even more ac-

ceptable than the reference translation. Table 5.2 shows two of such examples: *"scegli-este"* is a closer translation to *"would choose"* than *"poteste scegliere"* which mean *"could choose"*, or *"sto mostrando"* to *"I'm showing"*, in contrast to *"vi mostro"* (*"I show you"*).

As we have described in Section 3, a considerable section of the rule-based segmenter implementation is designed to deal with verb tenses. Although some frequent infections are merged back to the verb root with MergeBPE, Morph is able to segment with precision complex verbs such as the ones showed above (*"scegli §§ete", "scegli §§ereste", "scegli §§este", "mang §§erebbero", "mostr §§ando"*). We are thus confident to state that a better segmentation leads to a more grammatical translation in accordance with the results in Section 5.2.

5.3.3 Rare words and Out-of-Vocabularies

The use of subword units instead of full words in NMT was introduced with BPE by Sennrich et al. (2015) in order to deal with the open-vocabulary problem, ie. to overcome the problem in translating rare and out-of-vocabulary words in neural systems. One of the hypotheses at the beginning of our project was that with a morphological segmentation, an NMT model would have been able to separately translate the various morphemes and then combine them together. A process that would be extremely useful when dealing with inflections and morphological variations. In this section, we analyse how BPE and our morphological-based model dealt with such words.

For the non-morphological words, both BPE and Merge64K fail in translating them properly by removing the word from the output or returning a non-existing word as shown in the examples in Table 5.4. Nonetheless, Merge64K was able to effectively translate a large number of more complex words compared to BPE. Table 5.5 shows the translations of some complex words produced by the two systems. As we can see from the examples, a more morphological segmentation seems to facilitate the translations of such words, especially when there is a one-to-one translation of each morpheme as for the translation of the word "cryopreserve". We also noticed that instead of translating each subword, the morphological model preferred to transliterate most of the subword units and translate only the last morpheme. Although this might lead to a correct translation like for "cannibalizing" or "intransigent", it can also fail as for "preposterously" where, despite being correctly segmented, only "-ly" is correctly translated to "-mente".

System	Sentence
Src	[...] a spy thriller or a John Grisham novel.
Ref	[...] un thriller di spionaggio o di un romanzo di John Grisham.
BPE	[...] un romanzo di spiagge o un romanzo di John Grisham.
M.64K	[...] un romanzo di John Grisham.
Src	the ham and Swiss cheese omelets [...]
Ref	le omelette prosciutto e formaggio svizzero [...]
BPE	il prosciutto e i gioielli svizzeri [...]
M.64K	il prosciutto e lo scheletro svizzero [...]
Src	If I'm not a ladybug or a piece of candy [...]
Ref	Dopo, se non sono una coccinella o una caramella [...]
BPE	Se non sono un guaio o un pezzo di candela [...]
M.64K	Se non sono un furgone o un pezzo di candela [...]

Table 5.4: Examples of incorrect rare/OOV words translations from *tst2010* En→It for BPE and Merge64K. "[...]" indicates that the sentence shown is only a segment of a longer sentence.

In addition to the above, in Table 5.5 we show two more examples where our morphological-based model fails in translating complex words. "aribed" is not translated correctly by either model, however, the output provided by BPE can be considered more acceptable compared to Merge64K. This is a result of two problems, first of all, as we have discussed previously, our morphological-based model is not able to deal correctly with compounds. Secondly, it is affected by oversegmentation which injects noise into the translation.

To sum up, from our analysis it appears that a morphological segmentation does help to translate unseen words. However, a more rigorous study is needed in order to quantify its true abilities (for instance by creating a test set for such words).

5.4 Informal Human Evaluation

In the previous sections, we have found that our morphological-based models, compared to a BPE-based one, performed worse on MT automatic evaluation metrics. However, after a more in-depth analysis we believe that, despite some problems caused by oversegmentation, a morphological segmentation helps MT models in being more

System	Input	Output	Postprocessed
Src/Ref	cryopreserve them	-	criopreservarli
BPE	cr@@ y@@ opre@@ serve them	cri@@ ver@@ li	criverli
M.64K	cry@@ o## preserv §§e them	cri@@ o## preserv §§arli	criopreservarli
Src/Ref	cannibalizing	-	cannibalizzando
BPE	can@@ ni@@ b@@ alizing	can@@ ni@@ zz@@ ando	cannizzando
M.64K	can@@ ni@@ b@@ aliz §§ing	can@@ ni@@ b@@ alizz §§ando	cannibalizzando
Src/Ref	intransigent	-	intransigente
BPE	in@@ transi@@ gent	in@@ transi@@ zionale	intransizionale
M.64K	intra## n@@ sig §§ent	intra## n@@ sig §§ente	intransigente
Src/Ref	preposterously	-	estremamente
BPE	pre@@ po@@ ster@@ ously	estremamente	estremamente
M.64K	pre@@ post §§erous §§ly	pre@@ post §§erous §§mente	preposterousmente
Src/Ref	lesioned	-	lesionato
BPE	les@@ ion@@ ed	les@@ ion@@ ato	lesionato
M.64K	les@@ ion §§ed	les@@ sic §§o	lessico
Src/Ref	airbed	-	materassino
BPE	air@@ bed	letto d'aria	letto d'aria
M.64K	air@@ b§§ed	aere §§o	aereo

Table 5.5: Translations and segmentations of rare/OOV words in *tst2017* En→It for Merge64K and BPE. First half are correct translations for Merge64K while the second for BPE.

fluent and, potentially, translating a higher number of rare words.

Since it's practically impossible to evaluate a metrics such as fluency in an automatic way, we decided to conduct a small informal human evaluation in order to obtain the most unbiased judgement of the translation quality of the analysed systems. Furthermore, to keep the evaluation as simple as possible, we obtained results only for the BPE-based and Merge64K-based models.

5.4.1 Method

Given the limited resources at our disposal, the conducted evaluation has been realised through Google Form questionnaires. We made an evaluation for four translation directions[2]: two zero-shot (Romanian→Italian, Dutch→German) and two non-zero-shot

[2]Ro→It: https://goo.gl/forms/6rF6cMAm7S9VI8cM2
Nl→De: https://goo.gl/forms/BUyHHeYzg5EcBq3Z2
En→It: https://goo.gl/forms/GBAlHvPdM7K1CPnh1
It→En: https://goo.gl/forms/HXhA3W2cLoRMQaB93

Fluency	Adequacy
5: flawless	5: all meaning
4: good	4: most meaning
3: non-native	3: some meaning
2: disfluent	2: little meaning
1: incomprehensible	1: none

Table 5.6: Human evaluation metrics guidelines.

(Italian→English, English→Italian).

We prepared one Form per direction, each containing 100 sentences selected randomly from *tst2010* and *tst2017* (sentences for which the two analysed system produced the same output were discarded) and asked volunteers to participate in the evaluation[3]. To facilitate the evaluation task for the participants, we made the evaluation of each sentence non-mandatory and ask them to arbitrarily evaluate the sentences in the chosen Form and skip all the ones they were not sure about.

The evaluation criteria are the classical human evaluation metrics: fluency and adequacy. Each metric has to be evaluated with a ranking from 1 to 5 according to the guidelines in Table 5.6.

As shown in the example in Figure 5.2, each section is composed by:

- SRC: source sentence;
- REF: reference translations;
- Sys1/Sys2: translation generated by the BPE and Merge64K-based models. For obvious reasons, information about the systems which produced the translations was not given to the participant in the evaluation task.

5.4.2 Results

We kept the human evaluation Forms online for 6 days[4]. The statistics regarding the number of submission and sentences evaluated are shown in Table 5.7.

Despite receiving fewer evaluations than expected, we were able to find some useful insight into the Zero-Shot models. We first compare the rating obtained by the

[3]No personal information was requested for the evaluation.

[4]Actually, the forms are still online for "display" reasons. Probably it would've been more correct to say that the results have been collected after 6 days since their publication.

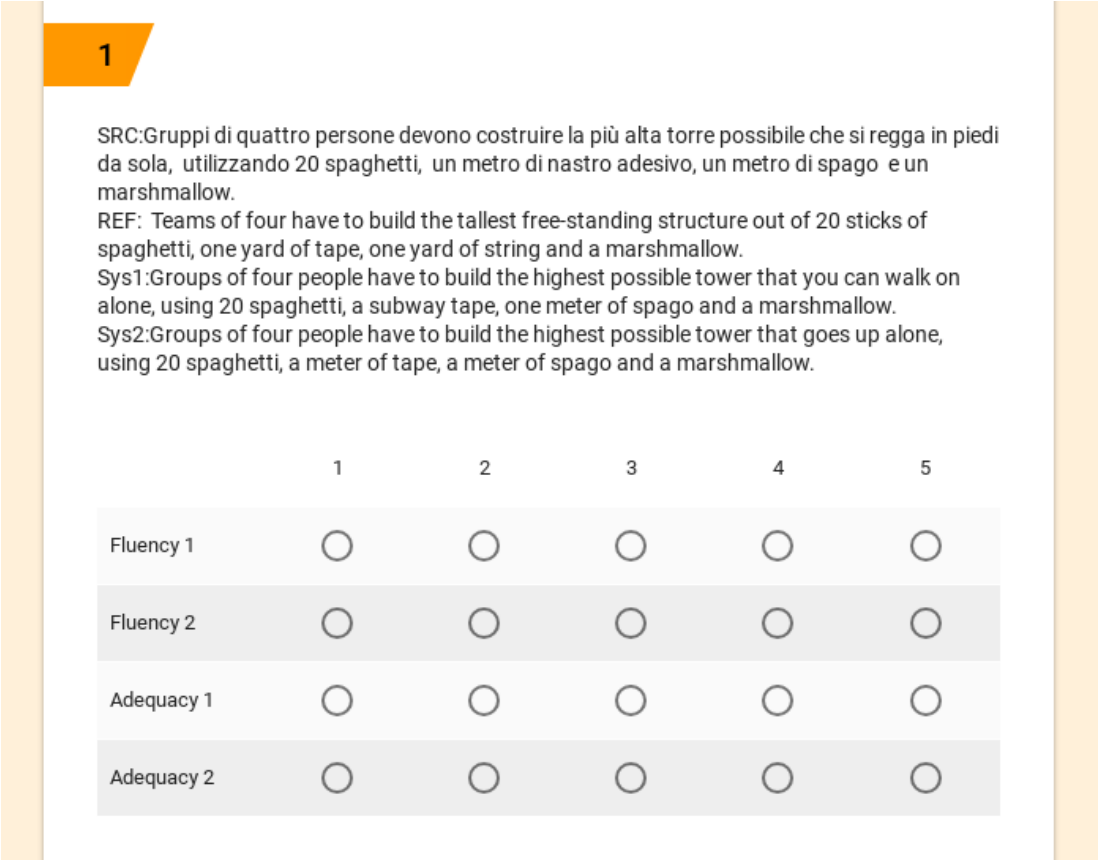
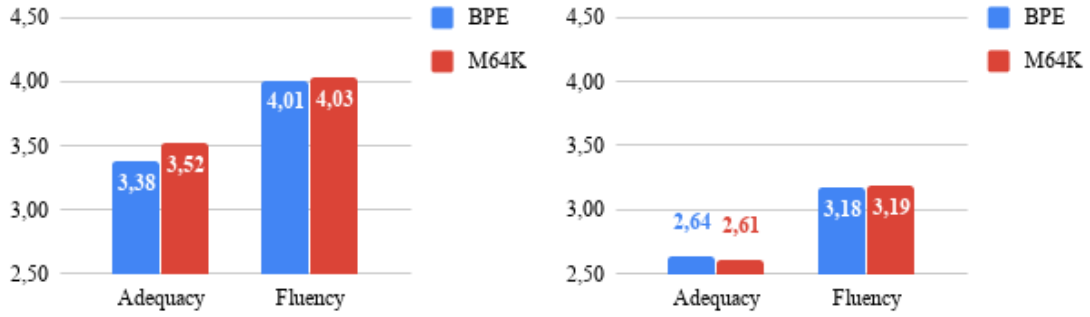


Figure 5.2: Example of form for a sentence in the Human Evaluation task. First sentence in the Italian→English Form.



(a) Non-Zero-Shot

(b) Zero-Shot

Figure 5.3: Adequacy and Fluency for Zero and Non-Zero-Shot directions.

various directions grouped by type (ie. the sums and counts of It↔En are merged to form the Non-Zero-Shot plot, while Ro→It and Nl→De's results are combined for the Zero-Shot one). As we can see from Figure 5.3, in accordance to what we have discussed in the previous analysis sections, a more morphologically-informed model does

Form	# Submissions	# Eval. sentences	Eval avg.
Ro→It	13	332	25
Nl→De	5	55	11
It→En	7	204	29
En→It	4	30	7
Total	29	621	21

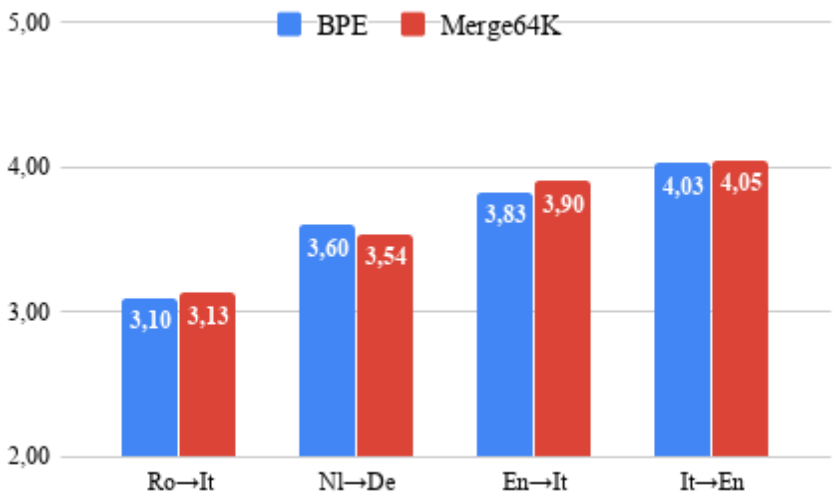
Table 5.7: Human Evaluation submission statistics. Number of submissions, number of evaluated sentences and average number of sentences per submission.

improve the fluency of an NMT translation. The improvement, although only slight, occurs for both Zero-Shot and Non-Zero-Shot translation directions. Furthermore, a morphological segmentation seems to help adequacy as well, however, a higher rating was achieved only for the Non-Zero-Shot. As we have discussed in Section 4.2.2, with lower resource conditions, the NMT model might be more susceptible to oversegmentation which nullifies both the benefits of the morphological and BPE segmentations.

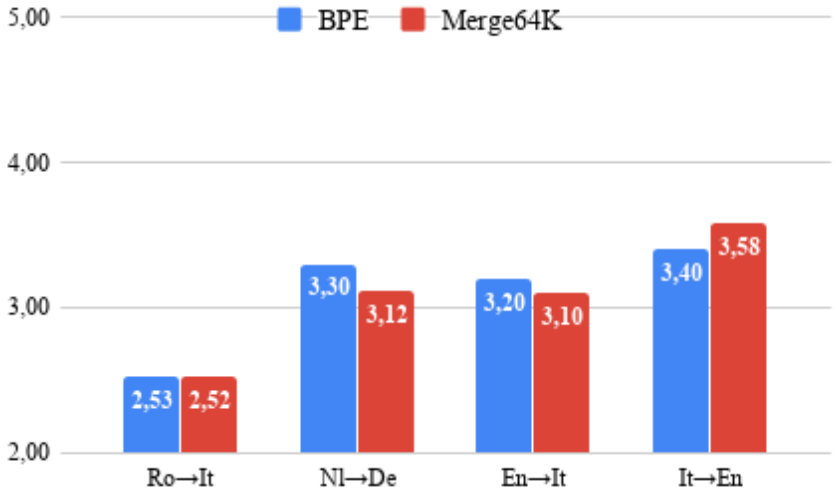
Figure 5.4 expands the results of the tables above for each translation direction. While the two systems performed comparably for all the translation direction, fluency and adequacy were quite different between the various languages. For both metrics, translations for It→En were the best ones. One of the possible reason is that translating to a morphologically-poorer language is easier compared to translating into a similar or richer one. In fact, we can see that the performance of the models on En→It are slightly worse with respect to It→En.

By looking at the Ro, En→It directions, we can see that, despite the fact that NMT models can able to translate between language pairs in zero-shot conditions, its performance drops considerably compared to a model with full training data. This result is in contrast with the results obtained by Dabre et al. (2017) where their zero-shot directions were comparable, if not better, to the models trained with complete parallel corpora. Further studies are necessary to compare the performances of RNNs and Transformers on Zero-Shot translations.

Among all the translation directions, the morphological-based model obtained a lower score in both metrics only for Nl→De. As pointed out by Huck et al. (2017b), German has an extremely high compound productivity. However, as we have noticed by manually examining translations in Section 5.3.3, Morph64K perform worse on compounds compared to BPE because oversegmentation prevents BPE from individu-



(a) Fluency

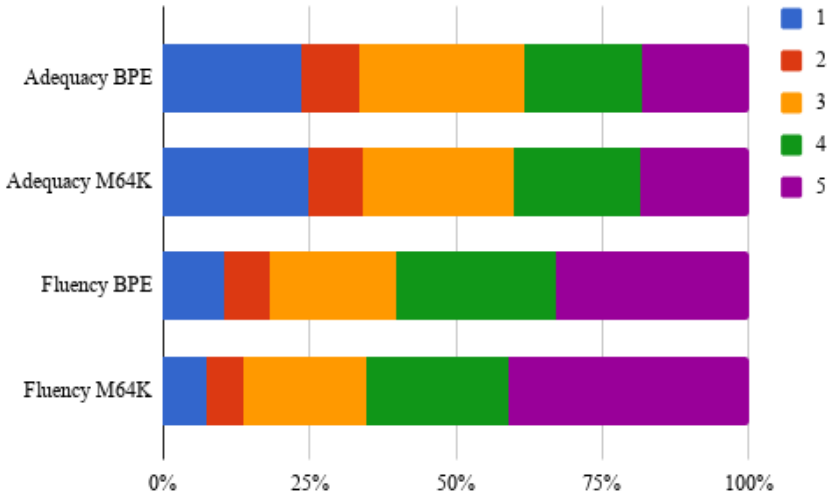


(b) Adequacy

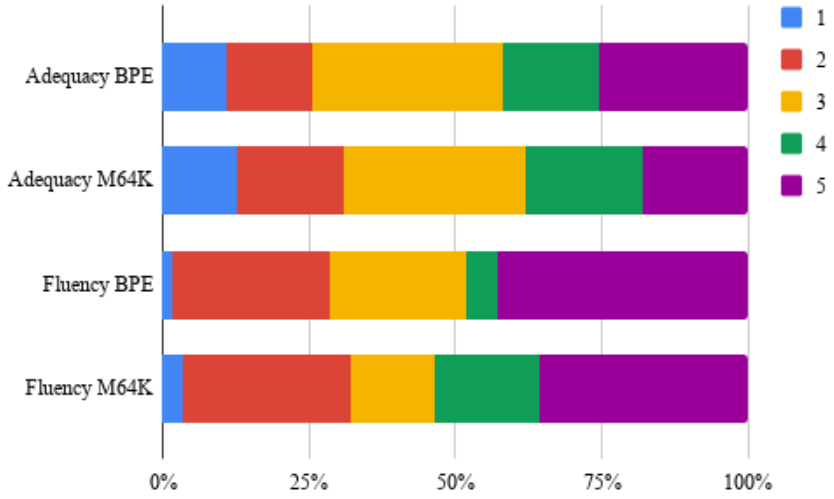
Figure 5.4: Average rating for human evaluation on four translation directions.

ating compound segments and thus translating them correctly.

As we can see from Figure 5.5, Merge64K's translations fluency for Nl→De obtained a lower number of 5 and 3 scores while increasing the numbers of 4 and 2 scores compared to BPE. On the other hands, the other examined Zero-Shot direction achieved a completely opposite result. Despite both directions works on the same conditions, for Ro→It, Morph64K outperforms BPE on fluency by obtaining a larger number of 5s and less 1s. What we can get from this is that the same segmentation technique may be more or less suitable for certain languages.



(a) Romanian→Italian



(b) Dutch→German

Figure 5.5: Human Evaluation results for zero-shot directions in percentage.

5.5 Discussion on MT automatic evaluation metrics

The best evaluation metric for MT is obviously Human Evaluation. Nonetheless, it is extremely rare to find studies where models were human evaluated if not on shared translation tasks due to the fact that human evaluation is expensive and time-consuming (Callison-Burch, 2009). For this reason, it is common to find papers where results and translation quality are expressed solely with automatic metrics. However, how good are MT automatic evaluation metrics in capturing actual translation quality?

Currently, the most wildly used and *de facto* standard evaluation metric for MT is

BLEU (Papineni et al., 2002). BLEU score is based on overlapping n-gram precision between the reference and output translation. In addition, the score is reduced for translations shorter than the reference through a *brevity penality*. BLEU score is good in the sense that it permits us to receive immediate feedback about the performance of an NMT model. Moreover, it can be considered a fair correlation metric to compare the translation to the reference due to the fact that it considers word order to some extent. However, this is at the same time one of the drawbacks of the metric. As Lee et al. (2016) pointed out "BLEU encourages reference-like translations and do not fully capture true translation quality".

Given a sentence, there are always multiple acceptable translations. Callison-Burch et al. (2006) defines BLEU as a metric that does not reflect the genuine improvements of an MT model since it "admits a huge amount of variation for identically scored hypotheses". As Graham et al. (2015) shows, even human annotators have a low agreement between each other, which means that human translators would perform poorly on BLEU score.

Disagreement between human and BLEU ranking can occur as shown by Wu et al. (2016). The most recent and relevant episode of disagreement between human and automatic evaluation occurred for the WMT'17 news translation shared task for English→German. The submission which obtained the highest BLEU score was Sennrich et al. (2017a) with 28.3 points, however, the human assessor of Direct Assessment ranked Huck et al. (2017b) higher compared to Sennrich et al. (2017a) despite obtaining only 27.1 BLEU score. Similarly, in this project, our morphological models achieved a worse performance if measured with BLEU compared to BPE, however, quantitative analysis showed that this might not be the case.

Chapter 6

Conclusions

6.1 Summary

The objective of this dissertation was to analyse the effect of morphology on Zero-Shot Neural Machine Translation systems. More specifically, whether a morphologically-driven segmentation strategy improves the translation quality of such systems. Our main claim was that a linguistically-informed segmentation would have helped an NMT model to better capture both internal and shared structures of the languages involved. In addition, we believed that the benefits would have been greater in Zero-Shot translations due to its extremely low-resource conditions.

In order to verify our hypotheses, we compared the performance of an NMT system trained with different segmentation techniques which can be grouped into three main strategies: BPE, rule-based segmentation and unsupervised morphological segmentation. The rule-based segmentation was implemented following the method proposed by Huck et al. (2017b) (which we called Morph for practical reasons) while we used Morfessor as an unsupervised morphological segmenter. In addition, we propose a new variation of the BPE algorithm to combine units instead of splitting them (MergeBPE). The purpose of this algorithm is to mitigate the oversegmentation problem that affects rule-based segmented words.

Our quantitative and qualitative analyses have shown that multilingual Zero-Shot NMT models indeed benefit from morphological information. The evaluation on contrastive pairs (Sennrich, 2016) showed that BPE is less grammatical in all the analysed error categories compared to our models. Additionally, our analysis showed that, despite not being able to fully transliterate proper nouns, the morphological models have some potential in translating complex tenses and rare/OOV morphological variations.

These results were confirmed by an informal human evaluation. In fact, compared to BPE, the evaluators ranked our morphological model higher for fluency on both Zero-Shot and non-Zero-Shot directions, while similarly for adequacy.

Despite these positive results, in all the automatically evaluated translation experiments, BPE resulted to be the best segmentation strategy for both the multilingual and Zero-Shot task. Morph and Morfessor segmented data, not only did not improve the model's performance but they reduced BLEU scores by $\approx$1 point on almost all translation directions. We argue that these contrasting results are due to the inability of automatic evaluation metrics to truly capture translation quality as pointed out by Lee et al. (2016). More specifically, such metrics assign high scores only to reference-like translations and do not consider variations or synonyms. However, thanks to these translation experiments we discovered some flaws in our morphological model, the first and foremost being oversegmentation. We have seen that by reducing the number of wrong morphological segmentation with MergeBPE, the performance of the models improve considerably, even with only a small number of merge operations.

6.2 Future work

First of all, although a rule-based segmentation algorithm is easy to implement, it is extremely hard (if not impossible) to implement a *perfect* morphological rule-based splitter. While it might be enough for extremely highly resource language pairs (Huck et al., 2017b), we have found that the loss caused by a wrong morphological word segmentation overpowers the benefits produced by a correct one, especially on Zero-Shot translation. For this reason, future work should primarily focus on achieving a better morphological segmenter. It is likely that a supervised segmenter would have sufficient precision to overcome the problems found in this project, however, it is hard to find such annotated data, especially for less-resourced languages. As for unsupervised morphological segmentation, we can improve the approach we used in this project by training the segmenter with higher quality data. In fact, our Morfessor model was able to achieve only 52.64% precision on the MorphoChallange2010. With better training data, Morfessor can achieve a precision up to 66.30% (Üstün and Can, 2016). Further experiments should also compare different unsupervised segmentation techniques. For instance, Passban et al. (2018) and Banerjee and Bhattacharyya (2018) obtained better BLEU scores over their baseline applying Morfessor *flatcat* (Grönroos et al., 2014). Another possible improvement is to use a shared subword separator token as suggested

by Banerjee and Bhattacharyya (2018).

Other experiments that we did not explore in this project might include more language-oriented experiments. In order to have a better understanding of the benefits of morphology in multilingual/Zero-Shot models, we should select more carefully the quality, quantity and languages of the dataset. For example, instead of having 5 languages, 3 might be more adequate to analyse the linguistic relations found by the NMT model (eg. the effect of a model with only languages from the same/different family).

Lastly, in this project, only fusional languages were involved. However, as shown by Ataman et al. (2017), morphological information have an extremely positive impact on NMT models that involve agglutinative languages. We believed that our approach might be effective in a similar way since a rule-based segmentation would be more precise and each morpheme less ambiguous.

Appendix A

Scripts and Additional Material

All the scripts, complete results and lists of affixes produced for this dissertation can be found at the following link:

https://github.com/GiulioZhou/Morpho_Zero-Shot_NMT

Here a list of the files:

- *morpho_segm*:
  - *removeTag.py*: remove XML tags and add target token;
  - *preprocess.sh*: preprocess from Morph to BPE. Multiple version of this file has been used obtained the various segmented data. *preprocess_test.sh* is for *tst2017*;
  - *morphoSegmenter.py*: given the language it segments the input file and saves it into the output. It calls the segmenter methods contained in the other files;
  - *suffixRemover.py*: NLTK-based morphological segmenter for German, Dutch and Romanian;
  - *itaSuffix.py*: Italian morphological segmenter developed from scratch;
  - *engSuffix.py*: English morphological segmenter developed from scratch;
  - *prefix.py*: rule-based prefix segmenter for the five languages
  - *prefixes*: list of prefixes for the five languages;
  - *affixes*: list of affixes for the five languages;
  - *learn_bpe_merge.py*: modified version of the learn BPE file for mergeBPE;
  - *apply_bpe_merge.py*: modified version of the apply BPE file for mergeBPE;

  - *morfessor_seg.sh*: segment a file with a Morfessor model. No training script is provided in this dissertation since the Morfessor models were trained with a simple command line that can be found in its documentation.

- *train_test*:

  - *concatenate.sh*: create source and target training and dev sets by concatenating all the files;
  - *concatenate_zero.sh*: create source and target training and dev sets by concatenating all the files but ignoring Zero-Shot directions;
  - *Various training files*: training files for Nematus. There are several training files however the base remains the same;
  - *translate.sh*: given a model and a test set, it produces segmented translations. *translate17.sh* is for translating *tst2017*.

- *evaluation*:

  - *results.xlsx*: excel file containing all the results obtained by the various model. In the files there are three spreadsheets for BLEU, chrF3 and other statistics. The higher results in the spreadsheets are results on *tst2010*, while the lowers are for *tst2010*. MULTI stands for multilingual model and ZERO for Zero-Shot models;
  - *postporcess.sh*: remove special tokens and detokenise the sentences; *computeBLEU.sh*: compute BLEU scores for all the language pairs; *computeCHRF3.sh*: compute chrF3 scores for all the language pairs;

- *others*:

  - *wordlist*: list of the words in the MorphoChallange testset;
  - *GS_seg*: list of morphologically segmented words in the MorphoChallange testset;
  - *get_frequency.py*: script composed of various functions: get the vocabulary of rare/unseen words; given a vocabulary and a file with sentences, returns a list of unseen word in the file and/or their respective sentence line; other methods for counting the number of rare words);

- *rare.sh*: given an output file produced by *get_frequency.py* and the corresponding source, translations and reference files, it return a file containing sentences grouped as source-ref-translation1-translation2.

Bibliography

Arad, M. (2006). *Roots and patterns: Hebrew morpho-syntax*, volume 63. Springer Science & Business Media.

Ataman, D., Negri, M., Turchi, M., and Federico, M. (2017). Linguistically motivated vocabulary reduction for neural machine translation from turkish to english. *The Prague Bulletin of Mathematical Linguistics*, 108(1):331–342.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Banerjee, T. and Bhattacharyya, P. (2018). Meaningless yet meaningful: Morphology grounded subword-level nmt. In *Proceedings of the Second Workshop on Subword/Character LEvel Models*, pages 55–60.

Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., and Glass, J. (2017). What do neural machine translation models learn about morphology? *arXiv preprint arXiv:1704.03471*.

Bender, E. M. (2013). Linguistic fundamentals for natural language processing: 100 essentials from morphology and syntax. *Synthesis lectures on human language technologies*, 6(3):1–184.

Berko, J. (1958). The child's learning of english morphology. *Word*, 14(2-3):150–177.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huang, S., Huck, M., Koehn, P., Liu, Q., Logacheva, V., et al. (2017). Findings of the 2017 conference on machine translation (wmt17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Logacheva, V., Monz, C., et al. (2016). Findings of the 2016 conference on machine translation. In *ACL 2016 FIRST CONFERENCE ON MACHINE*

TRANSLATION (WMT16), pages 131–198. The Association for Computational Linguistics.

Callison-Burch, C. (2009). Fast, cheap, and creative: evaluating translation quality using amazon's mechanical turk. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 286–295. Association for Computational Linguistics.

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluation the role of bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Casadio, C. and Lambek, J. (2001). An algebraic analysis of clitic pronouns in italian. In *International Conference on Logical Aspects of Computational Linguistics*, pages 110–124. Springer.

Cheng, Y., Yang, Q., Liu, Y., Sun, M., and Xu, W. (2017). Joint training for pivot-based neural machine translation. In *Proceedings of IJCAI*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Chung, J., Cho, K., and Bengio, Y. (2016). A character-level decoder without explicit segmentation for neural machine translation. *arXiv preprint arXiv:1603.06147*.

Creutz, M. and Lagus, K. (2005). *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0*. Helsinki University of Technology Helsinki.

Dabre, R., Cromieres, F., and Kurohashi, S. (2017). Kyoto university mt system description for iwslt 2017.

Dalvi, F., Durrani, N., Sajjad, H., Belinkov, Y., and Vogel, S. (2017). Understanding and improving morphological learning in the neural machine translation decoder. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 142–151.

Dryer, M. S. (2013). 26 prefixing versus suffixing in inflectional morphology.

España-Bonet, C. and van Genabith, J. (2017). Going beyond zero-shot mt: combining phonological, morphological and semantic factors. the uds-dfki system at iwslt 2017. In Sakti, S. and Utiyama, M., editors, *International Workshop on Spoken Language Translation*, pages 15–22. o.A.

Firat, O., Cho, K., and Bengio, Y. (2016). Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv preprint arXiv:1601.01073*.

Fortescue, M. (1984). Learning to speak greenlandic: a case study of a two-year-old's morphology in a polysynthetic language. *First Language*, 5(14):101–112.

Graham, Y., Baldwin, T., and Mathur, N. (2015). Accurate evaluation of segment-level machine translation metrics. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1183–1191.

Grönroos, S.-A., Virpioja, S., Smit, P., and Kurimo, M. (2014). Morfessor flatcat: An hmm-based method for unsupervised and semi-supervised learning of morphology. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1177–1185.

Gu, J., Hassan, H., Devlin, J., and Li, V. O. (2018). Universal neural machine translation for extremely low resource languages. *arXiv preprint arXiv:1802.05368*.

Ha, T.-L., Niehues, J., and Waibel, A. (2017). Effective strategies in zero-shot neural machine translation. *arXiv preprint arXiv:1711.07893*.

He, D., Xia, Y., Qin, T., Wang, L., Yu, N., Liu, T., and Ma, W.-Y. (2016). Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pages 820–828.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Huck, M., Braune, F., and Fraser, A. (2017a). Lmu munich's neural machine translation systems for news articles and health information texts. In *Proceedings of the Second Conference on Machine Translation*, pages 315–322.

Huck, M., Riess, S., and Fraser, A. (2017b). Target-side word segmentation strategies for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 56–67.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., et al. (2016). Google's multilingual neural machine translation system: enabling zero-shot translation. *arXiv preprint arXiv:1611.04558*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Koehn, P. (2017). Neural machine translation. *CoRR*, abs/1709.07809.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Koehn, P. and Knight, K. (2003). Empirical methods for compound splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 187–193. Association for Computational Linguistics.

Kurimo, M., Virpioja, S., Turunen, V. T., et al. (2010). Proceedings of the morpho challenge 2010 workshop. In *Morpho Challenge Workshop; 2010; Espoo*. Aalto University School of Science and Technology.

Lakew, S. M., Lotito, Q. F., Matteo, N., and Marcello, F. (2017). Improving zero-shot translation of low-resource languages. In *14th International Workshop on Spoken Language Translation*.

Lee, J., Cho, K., and Hofmann, T. (2016). Fully character-level neural machine translation without explicit segmentation. *arXiv preprint arXiv:1610.03017*.

Lieber, R. (2015). *Introducing morphology*. Cambridge University Press.

Lieber, R. (2017). Derivational morphology. *Oxford Research Encyclopedias*.

Lu, Y., Keung, P., Ladhak, F., Bhardwaj, V., Zhang, S., and Sun, J. (2018). A neural interlingua for multilingual machine translation. *arXiv preprint arXiv:1804.08198*.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Mauro, C., Marcello, F., Luisa, B., Jan, N., Sebastian, S., Katsuitho, S., Koichiro, Y., and Christian, F. (2017). Overview of the iwslt 2017 evaluation campaign. In *International Workshop on Spoken Language Translation*, pages 2–14.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Passban, P., Liu, Q., and Way, A. (2018). Improving character-based decoding using target-side morphological information for neural machine translation. *arXiv preprint arXiv:1804.06506*.

Pham, N.-Q., Sperber, M., Salesky, E., Ha, T.-L., Niehues, J., and Waibel, A. (2017). Kit's multilingual neural machine translation systems for iwslt 2017. IWSLT.

Pizzuto, E. and Caselli, M. C. (1994). The acquisition of italian verb morphology in a cross-linguistic perspective. *Other children, other languages: Issues in the theory of language acquisition*, pages 137–187.

Popović, M. (2015). chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Romera-Paredes, B. and Torr, P. (2015). An embarrassingly simple approach to zero-shot learning. In *International Conference on Machine Learning*, pages 2152–2161.

Sennrich, R. (2016). How grammatical is character-level neural machine translation? assessing mt quality with contrastive translation pairs. *arXiv preprint arXiv:1612.04629*.

Sennrich, R., Birch, A., Currey, A., Germann, U., Haddow, B., Heafield, K., Barone, A. V. M., and Williams, P. (2017a). The university of edinburgh's neural mt systems for wmt17. *arXiv preprint arXiv:1708.00726*.

Sennrich, R., Firat, O., Cho, K., Birch, A., Haddow, B., Hitschler, J., Junczys-Dowmunt, M., Läubli, S., Barone, A. V. M., Mokry, J., et al. (2017b). Nematus: a toolkit for neural machine translation. *arXiv preprint arXiv:1703.04357*.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Sennrich, R., Haddow, B., and Birch, A. (2016). Edinburgh neural machine translation systems for wmt 16. *arXiv preprint arXiv:1606.02891*.

Sestorain, L., Ciaramita, M., Buck, C., and Hofmann, T. (2018). Zero-shot dual machine translation. *arXiv preprint arXiv:1805.10338*.

Shapiro, P. and Duh, K. (2018). Morphological word embeddings for arabic neural machine translation in low-resource settings. In *Proceedings of the Second Workshop on Subword/Character LEvel Models*, pages 1–11.

Smit, P., Virpioja, S., Grönroos, S.-A., Kurimo, M., et al. (2014). Morfessor 2.0: Toolkit for statistical morphological segmentation. In *The 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Gothenburg, Sweden, April 26-30, 2014*. Aalto University.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Tamchyna, A., Marco, M. W.-D., and Fraser, A. (2017). Modeling target-side inflection in neural machine translation. *arXiv preprint arXiv:1707.06012*.

Üstün, A. and Can, B. (2016). Unsupervised morphological segmentation using neural word embeddings. In *International Conference on Statistical Language and Speech Processing*, pages 43–53. Springer.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Virpioja, S., Smit, P., Grönroos, S.-A., Kurimo, M., et al. (2013). Morfessor 2.0: Python implementation and extensions for morfessor baseline.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.