TÉCNICO LISBOA

Domain-Adapted Multilingual Neural Machine Translation

João Miguel Correia Alves

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. André Filipe Torres Martins
Dr. Amin Farajian

Examination Committee

Chairperson: Prof. Paulo Jorge Coelho Ramalho Oliveira
Supervisor: Prof. André Filipe Torres Martins
Member of the Committee: Prof. Bruno Emanuel da Graça Martins

January 2021

Acknowledgments

I would like first to thank my supervisors, André Martins and Amin Farajian. Furthermore, I would like to thank Unbabel, and in particular the machine translation team, for their help and cooperation. It was a wonderful experience, I was able to learn a lot about natural language processing and in particular neural machine translation.

I would also like to thank my family. A special thank to my mother, my father and my brother for always giving me the best support they could offer during this particular period.

Finally, I would to thank my friends. It would be impossible to finish this journey without their help.

Resumo

A Europa é um continente de diversidade linguística: A União Europeia tem atualmente 24 línguas oficiais. A globalização aumentou a necessidade de ter informação traduzida num maior número de línguas possíveis, o mais rápido possível. A tradução automática, e em particular a tradução automática neural, pode ser uma boa abordagem a este problema. A tradução automática neural proporciona condições ideias para o desenvolvimento de sistemas multilingue: torna possível a partilha de componentes para diferentes tarefas. Sistemas multilingue tornam possível o uso de um único modelo capaz de traduzir entre múltiplas línguas tanto na origem como no alvo.

Embora estes sistemas sejam bastante apelativos, ainda continuam a ter um desempenho inferior ao proporcionado por sistemas que suportam apenas um par de línguas, na maioria das situações. De modo a melhorar a performance dos sistemas que suportam múltiplas línguas, implementámos uma abordagem existente na literatura: *adapters*. *Adapters* são pequenas camadas residuais que são introduzidas no meio de modelos pré-treinados, que são usadas para adaptar o modelo para novas línguas, melhorando a sua performance. Procuramos melhorar este método introduzindo *adapters* que são condicionados em apenas uma língua (ao contrário da abordagem atual que condiciona num par de línguas). Fazendo esta mudança, torna-se possível extrair o potencial dos adapters para melhorar a performance, mesmo em cenários em que não existe informação paralela disponível.

Por fim, avaliamos empiricamente e comparamos a performance de sistemas que suportam múltiplas línguas e sistemas que recorrem a uma língua pivô em 24×23 pares de línguas. Inglês é a escolha habitual como língua pivô, mas nós procuramos estudar o uso de línguas pivô diferentes: Francs e Alemão, para transduzir entre línguas românicas e germânicas, respetivamente.

Palavras-chave: aprendizagem profunda, processamento de língua natural, tradução automática neural

Abstract

Europe is a continent of linguistic diversity: the European Union has 24 official languages. Globalization has increased the necessity of having information translated to many languages as possible, as fast as possible. Automatic translation, and in particular neural machine translation, can be a good solution to solve this problem. Neural machine translation provides an ideal setting for multilingual systems: it makes it possible to share components across multiple tasks. Multilingual systems make it possible to have a single model that translates from multiple source languages into multiple target languages.

While multilingual systems are appealing, the current reported performance is still behind that of dedicated bilingual models, for most language-pairs. To improve the performance of multilingual systems, we implement an existing approach in the literature, adapters. Adapters are tiny residual layers introduced in the middle of a pre-trained model that are used to adapt the model to a new language, improving its performance. We extend this method by conditioning adapters on one language only (as opposed to the language-pair setting initially proposed). By doing this, we are able to perform direct zero-shot translation and to improve the results in this scenario too.

Finally, we provide a thorough empirical analysis and comparison of different multilingual and pivot-based systems on 24$\times$23 language-pairs. While English is the usual choice of pivot language, we also study the use of different pivot languages, French and German, to translate between Romance and Germanic languages, respectively.

Keywords: Deep learning, natural language processing, neural machine translation, multilingual neural machine translation

Contents

List of Tables

List of Figures

Acronyms

RNN Recurrent Neural Network

LSTM Long Short-Term Memory

GRU Gated Recurrent Unit

NLP Natural Language Processing

SMT Statistical Machine Translation

NMT Neural Machine Translation

MNMT Multilingual Neural Machine Translation

BPE Byte Pair Encoding

BLEU Bilingual Evaluation Understudy

Chapter 1

Introduction

1.1 Motivation

Globalization has increased the necessity to have information translated in as many languages as possible. Nowadays it is possible to identify three different possibilities to approach the translation problem: human translators only, machine translation only, or a combination of machine translation with human post-editing. Although it is not on pair yet with human translators (Toral [1], Läubli et al. [2]), machine translation has proven to be very effective and useful in many applications while significantly cheaper and more scalable than human translator. Even in applications where quality is important, machine translation can be effective if used with human post-editing.

Early approaches to machine translation were rule-based and statistical (Brown et al. [3], Shannon and Weaver [4]). Statistical machine translation is the use of statistical models that learn to translate text from a source language to a target language given a large text corpus. These systems work by directly mapping a symbol or a subsequence of symbols in a source language to its corresponding symbol or subsequence in a target language. Usually, these approaches are specific to each language-pair and extending them to multiple languages requires significant engineering efforts. The first approaches were word-based. Later there was an improvement when the phrase-based approach was introduced.

Recently, a new generation of MT systems have emerged: neural machine translation (NMT) which ally the effectiveness and flexibility of neural networks with the increasing availability of data and computational power .The first approaches in this field were Recurrent Neural Networks (RNNs) using Long Short-Term Memory (LSTM) cells or Gated Recurrent Units (GRUs) (Sutskever et al. [5]). Later, a new mechanism was introduced in neural machine translation: the attention mechanism (Bahdanau et al. [6]). This mechanism allows the model to focus on specific words of the source sentence to generate the target word mimicking the idea of ”word alignment” present in IBM models (Brown et al. [3]). The transformer was introduced by Vaswani et al. [7] taking advantage of this mechanism. The idea behind the transformer is to handle the dependencies between input and output with attention completely.

The increasing computational capacity has allowed the emergence of multilingual systems (Dong et al. [8]). Neural machine translation makes it appealing to develop multilingual translation systems

since the neural architecture is language-agnostic and it is capable of capturing translation properties, such as long-distance re-ordering, even between highly dissimilar languages. It has already been shown that sharing a single translation model between multiple language-pairs can achieve competitive results when compared with strong bilingual baseline (Arivazhagan et al. [9], Aharoni et al. [10], Johnson et al. [11]), sometimes even with improvements. However, these improvements are not uniform: when translating to/from low-resource languages results may improve with transfer learning, but on the other hand high-resource languages are often penalized, due to the lack of capacity to accommodate all the language-pairs (Arivazhagan et al. [9]).

Existing approaches can be divided into shared models and language-specific encoder-decoder approaches. Johnson et al. [11] proposed sharing all parameters across all language-pairs, with one special token at the beginning of every source sentence indicating the target language. The main advantage of this approach is its simplicity and the fact that the number of model parameters remains constant. Language-specific encoder-decoder further divide into the ones that share parameters (Firat [12]) and the ones that do not share at all (Dong et al. [8]).

Although research in multilingual models has shown encouraging results, there are still some challenges:

- how to deal with a huge number of languages;
- how to deal with different scripting systems (vocabulary);
- how to deal with heavy imbalance data across languages and domains;
- how to define the practical limit on model capacity;
- how to deal with very heterogeneous inter-task relationships that come from dataset noise;
- how to deal with differing degrees of linguistic similarity.

Regarding the aerospace field, machine translation (and in particular multilingual neural machine translation) can have a tremendous impact. Although English is the language used in the aviation sector, it is not the native language of most of the world. Due to this, many maintenance errors have arisen (Drury and Ma [13], Drury and Marin [14]). Maintenance errors can be costly for both aircraft and human life. Simple misunderstandings can have a devastating impact. The aerospace field is facing a shortage of aviation maintenance technicians. While English is the language that is the most used in this field, the majority of the technicians speak English as a second language. Local languages can have an important role by helping to fill the shortage of technicians quickly. The new advances in the neural machine translation technologies have increased the speed and accuracy of translations and lowered the cost.

Furthermore, the European Space Agency (ESA) has 22 member states with different official languages. multilingual machine translation can take an important role to spread the work developed across the different countries and to facilitate mutual assistance between member states.

1.2 Contributions

The main contributions of this thesis are:

- We build multilingual NMT models handling the 24 European official languages, translating between any pair of languages among those 24 (while using only one model or a combination of two models). Our systems demonstrate good results, exhibit strong translation accuracy with much fewer parameters, improving the quality of low-resource languages (when compared with bilingual baselines), while keeping competitive results for high-resource languages;
- We inject tiny task-specific layers (adapters) into pre-trained models. The implementation is available online.[1] We provide in-depth analysis of various aspects of adapters that are crucial to achieve better quality in multilingual NMT. We demonstrate that is possible to close the gap to bilingual baselines with a small number of additional parameters;
- We provide a through empirical analysis and comparison among various strategies, including various choices of pivot languages.

1.3 Thesis Outline

Chapter 2 presents a brief introduction to the theoretical concepts needed for the development of this thesis.

Then, Chapter 3 covers a description of the state-of-art in multilingual neural machine translation and it presents our contributions. Afterwards, the experiments performed and the results obtained are discussed in Chapter 4. We present the bilingual baselines, and then the results with fully shared models. Then, we explore the use of adapters (tiny residual layers) to improve the performance of the multilingual systems. Finally, results achieved when using different pivot languages (English, French and German) are presented and analyzed.

Chapter 5 sums up the main achievements of this thesis and leaves suggestions for future work.

[1] Code available online at:https://github.com/JoaoMCAlves/Multilingual-Adapters

Chapter 2

Background

This chapter introduces the theoretical concepts that are needed to understand the work developed in this thesis. The focus of this chapter will be neural machine translation and some of the concepts associated to it. First, the existing neural machine translation models are presented. Then, a few concepts related with word representations are described. Finally, the BLEU score, a commonly used metric in NLP, is presented.

2.1 Neural Machine Translation

Neural machine translation is an approach to machine translation that takes advantage of the use of neural networks. In this section, it is going to be presented the neural networks that are usually used to perform neural machine translation and conclude which one is the best to serve our purpose.

2.1.1 Encoder-Decoder Approach

An Encoder-Decoder Network (Sutskever et al. [5],Cho et al. [15]) is the connection of two neural network, an encoder network and a decoder network. The encoder is responsible for encoding the input sentence into a context vector. The source sentence can be mapped into a fixed-dimensional vector or into a set of vectors. After that, the decoder is responsible for generating a target sentence based on this hidden representation.

In the field of neural machine translation, work focuses on recurrent neural networks (usually long short-term memories) and transformer models. In both cases, the encoder and the decoder have similar functions. In the next subsections, we explain these two models in detail.

2.1.2 Recurrent Neural Networks

Recurrent neural networks (RNN) are neural networks that have an ”internal” memory. RNNs are recurrent, because they perform the same steps for every input, while the output of the current input depends

on the past one that was calculated. After the output is calculated, it is sent back into the recurrent network. The RNNs are trained using an algorithm called backpropagation through time (Werbos [16]).

The current state (h_t) is a function of $\tanh$ of the previous state (h_{t-1}) and the current input (x_t):

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

where W_{hh} is the weight at previous hidden state, W_{hx} is the weight at current input state and $\tanh$ is used as an activation function to implement a non-linearity that squashes the values to the range -1 to 1. Finally the output is given by :

$$y_t = W_{hy}h_t$$

where W_{hy} is the weight at the output state.

Although RNNs can achieve good results, they present a few disadvantages. One commonly reported problem is that during training time the gradients become increasingly small throughout time steps (the vanishing gradient problem), making it hard for the model to capture long dependencies. To overcome this issue new solutions have arrived: Gated Recurrent Units (GRUs) (Chung et al. [17]) and Long Short Term Memories (LSTMs) (Hochreiter and Schmidhuber [18]). Another disadvantage of this architecture is that RNNs generate the hidden state based on the hidden previous steps states. This precludes parallelization within training examples. When the length increases this is an important limitation. transformers overcome this issue, as we will show in Section 2.1.3.

Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) networks are a modified (and more complex) version of RNNs, introduced by Rumelhart and McClelland [19]. They solve the problem of vanishing gradient. They have the ability of learning long-term dependencies. The structure of an LSTM cell is in Figure 2.1. LSTMs introduces three new components:

- **Forget Gate** - as the name suggests, this gate is responsible for discarding details that are not important to generate the next hidden state. It does that using a sigmoid function. Using the previous state (h_{t-1}) and the current input (x_t), it generates a number between 0 and 1 for each number in the cell state (C_{t-1}), where 0 means omitting it and 1 means keeping it, according to:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f)$$

- **Update/Input Gate** - this gate is responsible to decide which values from the input should be used to modify the memory. Firstly, a sigmoid function decides which values to let through. Then a $\tanh$ function gives weight to the values which are passed, deciding which are more important on

a scale that goes from -1 to 1.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i)$$
$$C_t = \tanh(W_C.[h_{t-1}, x_t] + b_C)$$

- **Output Gate** . this gate decides which part of the current cell should be in the output.

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \times \tanh(C_t)$$

With this three mechanisms, LSTMs are able to solve the issues of the original RNNs.

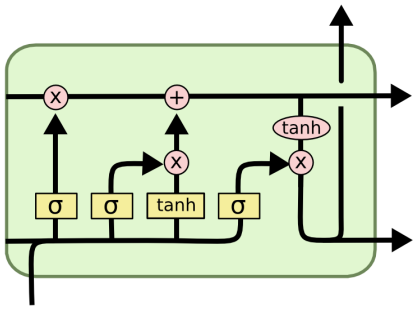

Figure 2.1: LSTM Cell.

2.1.3 Transformer

The transformer is considered the state of art for many tasks in natural language processing. Earlier models tended to have problems when decoding long sentences. To solve this problem the Attention mechanism was introduced. Attention (Bahdanau et al. [6]) is a mechanism that was developed to increase the performance of encoder-decoder models. At each time step, when the model tries to predict the next output word, the mechanism will focus on the parts of the input sentence where the most relevant information is. The transformer is a model that takes advantage of this mechanism to improve the training speed and the performance of natural language processing tasks, in particular neural machine translation.

In this subsection, we explain how it is implemented. First, we explain how the attention mechanism works. After describing this, we are ready to explain the components of the transformer: the **embedding layer**, the **encoder** and the **decoder**. Figure 2.2 depicts the transformer architecture.

Attention Mechanism

The self-attention or scaled dot-product attention allows the model to associate each individual word in the input to other words in the input. To perform the self-attention, the input is feed to three different

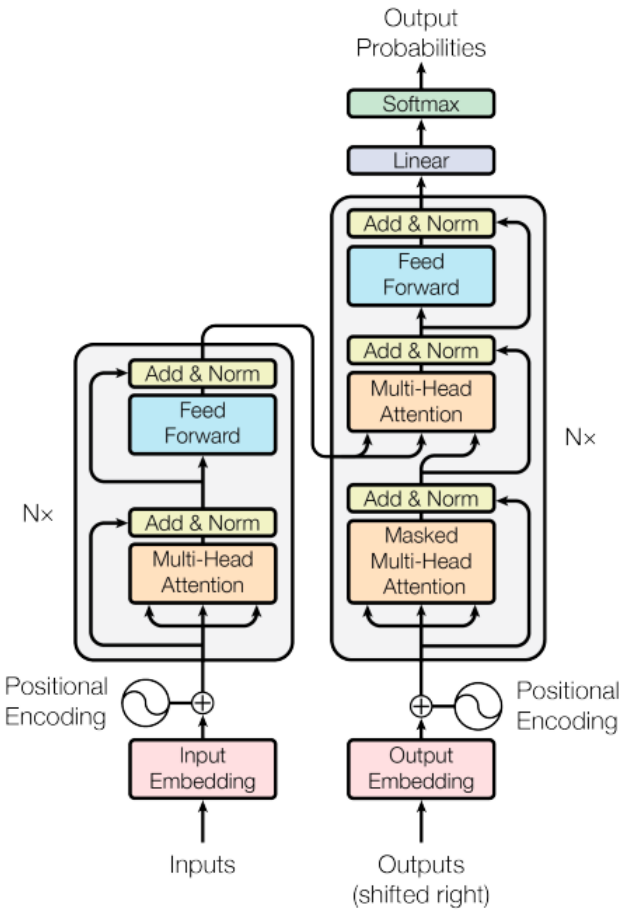

Figure 2.2: Transformer Architecture. Taken from Vaswani et al. [7].

linear layers to create three vectors: **key** (K), **query** (Q) and **value** (V).

Then, the query and key vectors are multiplied to produce a score matrix that indicates how much focus a word should put on the other words, so each word will have a score that corresponds to other words. The higher the score, the higher is the focus given to that word. Then, these scores are divided by the square root of the dimension of the keys (or the queries or the values because they have the same dimension). This is done just to allow stable gradients as multiplying can lead to exploding values. Then a softmax function is applied, to have values between 0 and 1. Furthermore when softmax is used, higher scores are heightened and lower scores are depressed which allow the model to be more confident on which words should take attention to. Then, the attention scores are multiplied by the value vector to get an output vector. The attention-mechanism can be described by the following equation:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Each layer are used multiple self-attention heads. This is done because each head can learn different features, giving the encoder a higher representation power.

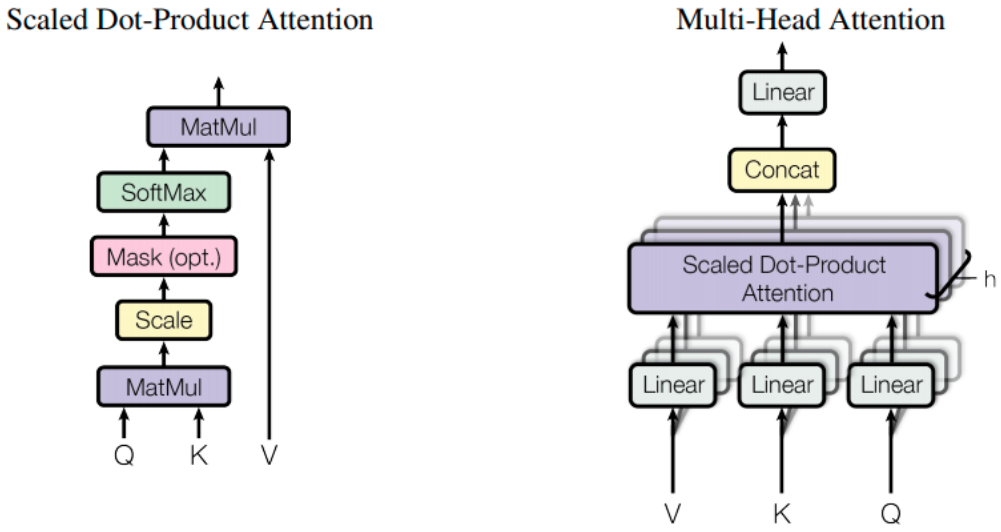

Figure 2.3: Transformer Attention Mechanism.Taken from Vaswani et al. [7].

Embedding

The first step is feeding our input into a word embedding layer. A word embedding layer can be seen as a table where each word is represented by a vector of continuous values. The next step is injecting positional information into the embedding. This is done because the transformer encoder does not have recurrent connections as RNNs, so we need to add information about the positional embedding. This is done using trigonometric functions (sine and cosine). These functions are used because they have linear properties that the model can learn easily to attend to. The authors proposed to create a vector using the cosine function for every odd index on the input vector, and a vector using the sine function for every even index. Then, they proposed to add these vectors to their corresponding input embeddings. This gives the necessary information on the position of each vector.

Encoder

As described in the previous section, the encoder is responsible for mapping all the input sentence into an abstract continuous representation that holds the information of the entire sequence.

The encoder is composed of a stack of layers (the original paper uses six layers, but this number is configurable). The layers are all identical to each other, but they do not share the weights. Each layer has two sub-layers, a **self-attention layer** (also called multi-head attention) followed by a **feed-forward layer**. There are also residual connections around each of the two sub-modules followed by a layer normalization. The residual connections allow the gradients to flow through the network directly and the layer normalizations are used to stabilize the network which reduces the training time.

First, the output of the embedding layer is fed to the multi-head attention. Then the multi-head attention output is added to the original which is called a residual connection. The output of the residual connection goes through a layer normalization.

Finally, the normalized residual output is fed to a feed-forward linear network for further processing. This feed-forward linear network is just a combination of two linear layers with a *ReLU* (rectified linear unit) activation function in between them. The output of this linear is again added to the input and again normalized.

The transformer uses more than one layer because by doing that, it allows different layers to learn different attention representations.

Decoder

The decoder takes the continuous vector representation that is given by the encoder and generates the output step by step, while is also fed previous output decoder recurrently, until an ”end of sequence” token is generated. As in the case of the encoder, the decoder is also a stack of layers (it has the same number of layers as in the case of the encoder).

The decoder has similar layers as the encoder. Each layer has two multi-head attention layers and a feed-forward layer with residual connections and layer normalization after each sub-layer. The two multi-head attention layers have different functions.

The decoder takes its previous outputs as inputs as well as the encoder outputs that contain the attention information from the input. The input goes through a positional embedding layer. The positional embeddings are fed into the first multi-head attention layer which computes the attention score for the decoder input.

This multi-head layer has a different function. Since the decoder is an auto-regressive network, it is important not to look into future tokens/words. Masking is used to prevent the decoder to look into future words. This mask is called look-ahead mask. The mask is ahead before calculating the softmax. The mask is a matrix of the same size of the attention scores that is filled with zeros and negative infinities. The right triangle is filled with negative infinities which will lead to zeros when the softmax is calculated. By doing this, we are forcing the model not to look at future words. The output of this first multi-head attention has information about how the model should focus on the decoder’s input.

The output of the second multi-head attention layer goes through a point-wise feed-forward layer for further processing. Then it goes through a final linear layer that acts as a classifier. The output of the classifier is fed into a softmax layer. The softmax layer will produce a number that goes from 0 to 1 for every word. It corresponds to the probability of the word generated. Finally, the transformer takes the index of word with higher probability and that is the predicted word.

2.1.4 Comparison of Transformer and RNNs

After having presented the different architectures, it is important to understand which one can achieve better results in neural machine translation. The work developed by Lakew et al. [20] has shown that transformer achieves better results than RNN in most of the translation tasks. They have tested both models in bilingual models, multilingual models and zero-shot translation. Although recent versions of RNNs as LSTMs have a longer reference window. In theory, if the computational power was infinite,

transformer would have the ability to have an infinite reference window. Moreover, they proved that in multilingual models, when using transformers, the transfer learning is more effective than in the case of using RNNs.

2.1.5 Word Representation

Neural networks deal with numerical data. Because of that, it is necessary to preprocess words and represent them as vectors. It is possible to identify two possible approaches to this problem: **one hot encoding** and **dense feature embedding**.

With one hot encoding, each vector has same size as the vocabulary. We would have a vector of zeros except for the cell at the index representing the corresponding word in the vocabulary. Although this approach is very easy to implement, it does not allow to extract relations between words. The representations are all going to be independent of each other.

The alternative is to use dense feature embeddings. While vectors obtained with one hot encoding are binary, word embeddings are low-dimensional floating-point vectors. Words that have a stronger relation between them will be closer in the representation space. The other main advantage of dense feature embedding is that it requires less computational power. Usually, the dimension of the vector that represents each word is inferior to the dimension of the vocabulary. This is not possible to obtain in one-hot encoding as the word representations are all independent.

2.1.6 Vocabulary

To generalize to unseen sentences, it is necessary to decompose these sentences into a set of basic units. In natural language processing, this set of basic units is called the vocabulary. Given a new text, it is preprocessed and decomposed into its basic units. A well-defined vocabulary needs to fulfil two conditions:

- produce a minimal number of *unknown* tokens, which means to have a high coverage;
- have a reasonable size to limit computational cost.

The first NMT models used word-level approaches (Bahdanau et al. [6], Cho et al. [21]). Some issues arose due to the difficulty of capturing all words within a limited vocabulary. To solve this issue, character-level systems were then proposed (Wang et al. [22], Ling et al. [23]). This approach solved the problem of capturing all words but increased the computational cost due to the increased sequence lengths. Subword level vocabularies (Sennrich et al. [24]) are the ones that achieve a good balance between the coverage and the computational cost.

Byte Pair Encoding (BPE)

As explained above, subword level vocabularies achieve a good balance between the coverage and the computational cost. Byte Pair Encoding (BPE) (Sennrich et al. [24]) is a common algorithm to create

subwords. This approach is based on the idea that different words share some parts (units) which the translation can be obtained by concatenation of this subwords. It is particularly important in languages where there are a huge number of compound words (e.g. German).

After having the corpus preprocessed, the first step to use this algorithm is to define the desired vocabulary size. Then, it is necessary to split words to sequence of characters and to append the suffix "@@" to the end of the word with its frequency. We begin with subword units that correspond to the characters. After this, a new subword is generated which corresponds to the highest frequency occurrence. The last step is repeated until we reach the desired vocabulary size or until the highest frequency occurrence corresponds to 1.

To conclude, BPE gives us a good balance between character level and word level representation. It is a good choice to deal with large corpora.

2.1.7 Evaluation Metrics

Metrics for automatic evaluation have been very important for the rapid progress of machine translation. Before their creation, it was necessary to rely on the work of human annotators which is a slow and expensive process.

BLEU Score

One of the most used evaluation metrics is the BLEU (Bilingual Evaluation Understudy) score. This metric is quick and inexpensive in terms of computational cost and it correlates highly with human evaluation. Furthermore, it is easy to understand and it is language independent.

A perfect result corresponds to 1.0 BLEU, whereas a complete mismatch corresponds to 0.0 BLEU. It is usually reported on a scale from 0 to 100.

But how is it calculated? As it is described in Papineni et al. [25] BLEU may be defined as:

$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.1}$$

where BP is the brevity penalty (used to penalize translation shorter than the references, n the number of n-grams and p_n the n-gram modified precision. By default N is equal to 4 and $w_n = \frac{1}{N}$.

The brevity penalty can be calculated as:

$$BP = \begin{cases} 1, & if c > r \\ \exp(1 - \frac{r}{c}), & if c < r \end{cases} \tag{2.2}$$

where c is the length of the candidate and r the length of the reference.

When the machine translation is identical to the reference, BLEU is 1.0. That is the reason why even a human translator usually does not obtain 1.0. It is possible to have a correct translation that does not achieve a high BLEU score. This means that a good translation does not imply having a BLEU score of 1.0.

2.2 Summary

This chapter introduced the necessary background for this master thesis. First, it introduced the models that are used in Machine Translation: RNN (and in particular LSTMs) and the transformer. Then, we described why the transformer is the usual choice nowadays. The attention mechanism, a powerful tool, makes it possible to capture long term dependencies.

Next, we presented a few concepts related to word representations in NLP. In particular we explained the importance of creating a good vocabulary and explained what are the current approaches in multilingual NMT.

Then, BLEU was presented. BLEU is one of the most used metrics in neural machine evaluation tasks.

Chapter 3

Multilingual Neural Machine Translation

Multilingual translation models are systems that share a single translation model between multiple language-pairs. According to the language-pairs and translation directions chosen, the multilingual neural machine translation (MNMT) systems can be used in different configurations:

- **many-to-one** - multiple source languages and one target language. This can be described as a multi-domain problem;
- **one-to-many** - one source language and multiple target languages. In this case, we have a multi-task problem;
- **many-to-many** - multiple source and target languages.

In this chapter, we present the state of the art approaches for multilingual neural machine translation. First, we present the overall architectures that these systems can have: **multilingual systems with language-specific encoders and decoders** and **universal encoder-decoder systems**. Then we explore one of the main advantages of using multilingual NMT systems: **transfer learning**. Especially, we focus on an extreme case of transfer learning: **zero-shot translation**. We further divide zero-shot translation into two sub-categories: pivot-based zero-shot translation and direct zero-shot translation.

Afterwards, adapters are presented. Adapters are residual layers introduced in a pre-trained model. Bapna and Firat [26] was the first work to use them in multilingual neural machine translation. We extend their work by introducing some changes that further improves their performance. Our approach allows us to use adapters even in direct zero-shot translation.

3.1 Overall Architecture

One advantage of multilingual models is the number of models (and consequently the number of parameters) that they require. If we want to translate between N languages, if we follow a naive approach and use individually trained models, it would require $N \times (N-1)$ models. If N is too large, it will be

impractical to deploy and maintain this huge number of models. Multilingual approaches reduce the number of parameters required: depending on the approach chose the number of parameters can be constant (**Universal Encoder-Decoder Models**) or it can grow linearly with the number of languages, $O(N)$ (**Models with Language-Specific Encoders and Decoders**). In this subsection, we present these two different approaches.

3.1.1 Multilingual models with Language-Specific Encoders and Decoders

Language-specific approaches, as the name suggests, require specific encoders or decoders for each language. Some additional features are added to produce shared representations. Although they lead better with the problem of accommodating more language-pairs, they do not take full advantage of the transfer learning feature. Furthermore, the training process tends to be slower and there is an increase in memory requirements due to the increase in the number of parameters.

The first multilingual neural attempt was proposed in Dong et al. [8]. The authors proposed a one-to-many model with a single encoder but separate decoders and attention mechanism for each target language. The study has hown that it was possible to improve the results of low-resource languages by using a mix of low-resource languages and high-resource languages. This architecture was able to make full use of the source language data (English) across different language-pairs. Lakew et al. [27] proposed a similar approach.

Then in Luong et al. [28], the authors proposed a many-to-many model with multiple encoders and decoders, one for each source and target language. Here, the different models shared the recurrent connections and the corresponding embedding space (single shared vector space) with a fixed length representation across all source and target pairs. The authors did not use an attention mechanism. As we mentioned earlier, this kind of approaches does not take full advantage of positive transfer learning (sharing information across similar languages) which may result in worse results for low-resource languages.

Firat et al. [29] proposed a many-to-many model (with up to 6 languages) with language-specific encoders and decoders with a single attention mechanism. This was the first work to introduce the idea of direct zero-shot translation. Once again, they showed improvements in low-resource settings. The authors argue that they may use, for each language, encoders and decoder with different architectures or different sizes. Vázquez et al. [30] had a similar approach to the multilingual problem. The study proposed language-specific encoders and decoders but with a shared independent attention bridge. The authors did experiments using many-to-one, one-to-many and many-to-many models.

In Zoph et al. [31], the core idea was to train a high-resource language-pair (parent model) and then copy the parameters to a low-resource model (child model). The authors found they could achieve better results by fixing certain parameters of the parent model and fine-tune the rest with the child model. They also used an attention mechanism that allows the target decoder to look back at the source encoder.

Lu et al. [32] proposed the idea of an explicit *neural interlingua*: an intermediate language that is able to represent all languages embeddings. The *neural interlingua* receives language-specific encoder

embeddings and produces output embeddings which are agnostic to all languages. Each language has its specific encoder and decoder,

Escolano et al. [33] proposed an alternative to introducing new encoders and decoders per language without retraining the entire system. The study propose to use language-specific encoders and decoder with a common intermediate representation space. The authors do not share any parameter across encoders or decoders. They initially train a set of language-specific encoders and decoders. Then, if they want to add new languages, they only need to train the new encoders/decoders.

3.1.2 Universal Encoder-Decoder Models

A universal encoder-decoder model uses only one encoder and one decoder for all language-pairs. The main advantage of this approach is its simplicity: it does not require special components or any special attention mechanisms. It allows integrating any language in the source or target side of the encoder-decoder architecture with only one encoder and one decoder. Moreover the model can achieve good results with a much smaller number (and constant) of parameters. A universal encoder-decoder model is able to accommodate all translation directions within a single model which reduces the training time and simplifies the processes of deployment and maintenance. However, the model capacity is a strong limitation to this kind of approaches. It has been shown that increasing the capacity is directly related with better results, but scaling capacity leads to a significantly larger computational footprint. Moreover, if we want to add a new language or new data, the whole system needs to be retrained and the quality of translations drops when we add too many languages, especially for those with the most resources (Arivazhagan et al. [9]). This problem is more evident when languages are not from the same language family. However, it is important to notice that naively increasing the model capacity might result in poor transfer performance to low-resource languages. Arivazhagan et al. [9] and Zhang et al. [34] claim that the model capacity is one of the most important factors, so it is crucial to find the optimal capacity for our system.

Johnson et al. [11] and Kudugunta et al. [35] proposed a many-to-many model that shares all parameters across all language pairs. To do that, the authors used a shared vocabulary for all languages in the dataset. They only introduced a special token at the beginning of every source sentence indicating the target language. The authors expected to obtain good translation results mainly when the target languages are related. The authors were able to perform direct zero-shot translation without any special treatment.

Further work proposed sharing all parameters except the attention mechanism, showing improvements over sharing all parameters (Ha et al. [36] and Blackwood et al. [37]). Ha et al. [36] added a special token at the beginning and at the end of every source sentence indicating the target language.

In Blackwood et al. [37], the authors tested adding different tokens at the beginning of their source sentences: target specific, source specific and pair specific. The study had the best results when target-specific tokens were used. The authors performed their experiments using four languages from the Europarl corpus (Iranzo-Sánchez et al. [38]). They achieved better quality translation for all possible

translation directions, compared to a model which shares all parameters. Moreover, they have tested three different attention mechanisms: target-specific attention, source-specific attention and paired attention which represents a specific language combination. In each situation, each attention mechanisms has a set of attention weights and bias parameters. The best results were achieved with a target-specific attention model.

Recently, Platanios et al. [39] also proposed to share the entire network but introduced a new mechanism: Contextual Parameter Generator (CPG). This mechanism learns to generate the parameters that the system should use based on the source and the target languages. The contextual parameter generator receives the source and the target language embeddings as inputs, and generates the parameters for the encoder and for the decoder (LSTMs networks). The rest of the model parameters remain unchanged and are shared across all languages. The authors treat language as context with which to encode or decode. They embedded languages in separate vector spaces.

3.1.3 Vocabulary

The construction of the vocabulary is a critical factor when developing a new massively multilingual translation system due to the high number of different character sets and morphological variance. It is possible to divide the existing approaches into three categories:

- **Separate vocabularies** for each language (Dong et al. [8], Luong et al. [40], Firat et al. [29])
- **Shared vocabularies** for all languages (Post et al. [41], Ha et al. [36], Johnson et al. [11])
- **Hybrid solutions** (Lakew et al. [27])

When constructing a **shared vocabulary** there are some problems that may arise: what should be its dimension and how should the vocabulary be created not to favor some languages over others due to the imbalance of the dataset size. Moreover, this method relies on the idea that the languages share a good number of word pieces. If this is not the case, then either the decoder's output layer will be very large (which slows the training) or we can have too many *unknown tokens*. Arivazhagan et al. [9] has explored the effect of varying the vocabulary size for a massively multilingual model. The authors concluded that the shared vocabulary size grows dramatically when we add a large number of languages (especially if they do not use the same scripts).

On the other hand, if the choice is having **separate vocabularies** for each languages, these problems do not arise. The only question that is necessary to take into account is the number of vocabularies that we are going to deal with. If the choice is to build a massively multilingual NMT system, it is necessary to deal with a huge number of vocabularies.

Lakew et al. [27] tried to have the best of both worlds with an **hybrid approach**. The authors proposed to add new languages to a system by adapting the vocabulary. They started with a shared vocabulary constructed with the initial languages available. When they added new languages, they adapted their vocabularies (dynamic vocabulary).

Regarding the architecture choice, if we use a universal encoder-decoder approach it is necessary to have a shared vocabulary. If the choice is a language-specific encoder-decoder approach, this archi-

tecture allows each language to have its own vocabulary because they are trained in parallel but it is possible to have a shared vocabulary too.

3.2 Transfer Learning and Zero-Shot Translation

Transfer Learning is the mechanism that enables the knowledge from a learned task to improve the performance on a related task, which typically reduces the amount of data needed to achieve the same results. In Natural Language Processing (Pan and Yang [42], Ruder [43]), transfer learning has already been applied to different tasks such as speech recognition (Kunze et al. [44]), document classification (Lu et al. [45]) or sentiment analysis (Dong and de Melo [46]).

In multilingual NMT, low-resource languages take advantage of being trained together with high-resource ones. This mechanism is even stronger across similar languages. The most common technique consists of training together low-resource and high-resource languages. Zoph et al. [31] tried a different approach. The authors used a pre-trained model (parent model) with a high-resource language-pair and then copied the parameters to the model that is going to be used to translate the low-resource language-pair (child model). By doing this, they reduced the performance on the low-resource languages too.

An extreme case of transfer learning is zero-shot translation. In this case there is no parallel data between the languages that we are considering. It is possible to divide zero-shot translation into two categories: **direct zero-shot translation** and **pivot-based zero-shot translation**.

3.2.1 Pivot-Based Zero-Shot Translation

The most common alternative to multilingual systems is pivot-based zero-shot translation using bilingual direct models. It is a useful method for translating between languages with little parallel data by utilizing parallel data in an intermediate language which usually is English. As English is by far the language with the largest volume of parallel data, it is an easy choice. The text is firstly translated from the source language to the pivot language, and then from the pivot language to the target language. In this process it is necessary to use two different systems, a source-pivot one and a pivot-target one.

Although this strategy usually achieves good results, it has a few disadvantages. The two-step translation strategy has the potential to propagate errors. As it is a two-step translation system, it requires doubling the latency and computational overhead (due to translating the source sentence twice) which is a concern for large-scale NMT models. Moreover, there is the possibility of losing important information when the source is translated to the pivot language.

It is very common to use bilingual direct models to perform pivot-based zero-shot translation. In this master thesis, we are going to combine a multilingual approach with pivot-based zero-shot translation. First, we are going to train a many-to-one multilingual model and a one-to-many multilingual model. Then we are going to use them to perform pivot-based zero-shot translation, using the many-to-one model followed by the one-to-many model. We aim to achieve similar performance results or even better

than the ones obtained with bilingual models for pivoting.

Use of different Pivot Language

English is the usual choice as pivot language. English is a popular language due to the parallel corpora available. However, there are factors such as language relatedness that can affect the choice of the pivot language for a certain language-pair.

For example, if we want to translate from Spanish to Portuguese, we could first translate from Spanish to English and then translate from English to Portuguese. As an alternative, we could use French as the pivot language, translating first from Spanish to French and then from French to Portuguese. French, Spanish and Portuguese are all Romance languages, they belong to the same language family. Would the results be better? In this master thesis, we explore the use of two different pivot languages: French (fr) and German (de). We provide comparison between these two pivot languages and English (Section 4.8). We use French as pivot language to translate between Romance languages and German as a pivot language to translate between Germanic languages.

3.2.2 Direct Zero-Shot Translation

Direct zero-shot translation does not require the intermediate step of translating into a pivot language. The multilingual system is trained with multiple source and target languages, and it has the ability of translating between them. Although pivot-based zero-shot translation yield higher BLEU scores than direct translation, recent works (Zhang et al. [34], Arivazhagan et al. [9]) suggest that in the near future, direct zero-shot translation is going to be able to perform as good or better than pivot-based zero-shot translation. New techniques and the increasing computational power have been responsible for that.

Firat [12] was the first to attempt direct zero-shot translation. They tried to perform it with several encoders and decoders but the results were very poor. The authors concluded that it requires an additional fine-tuning step. They used synthetic training data generated through pivoting to train translation directions that do not have parallel data available.

Escolano et al. [33] also performed direct zero-shot translation using a language-specific encoder-decoder architecture but the results were behind the ones achieved with Universal-Encoder approaches. Universal Encoder-Decoder approaches have demonstrated the ability of translating between any language-pair, without using pivot languages and without any special treatment. The shared representation space across languages induces transfer learning. In Johnson et al. [11], the authors obtained reasonable results but they tested their direct zero-shot systems on related languages and large-scale datasets. Blackwood et al. [37] was able to improve translation quality even in low-resource translation directions. Other works (Chen et al. [47], Currey and Heafield [48]) generate pseudo-parallel data using pivot-language monolingual data and incorporated it into the training process.

One problem associated to direct zero-shot translation is the off-target translation issue. It is very common to identify translations into a wrong language (usually English, especially if an English-centric dataset is used). In Section 4.7.2 we show some examples of this phenomenon. Zhang et al. [34] pro-

posed three different mechanisms to overcome this issue: use of layer normalization conditioned on the target language, use of a linear transformation between the encoder and the decoder also conditioned on the target side and introduction of a new backtranslation algorithm (ROB_T, which stands for Random Online Backtranslation).

3.3 Adapters

Adapters are tiny residual layer that are introduced in the middle of a pre-trained model to improve its performance. This approach shares a large set of parameters across all tasks and introduces a small number of task-specific ones. Adapters have been introduced as an alternative to fine-tune all weights of the pre-trained model.

The main advantage of adapters is that they do not require full fine-tuning of all parameters of the pre-trained model. The pre-trained model can be trained ahead of time and then the adapters can be introduced. Once the adapters are introduced, the parameters of the pre-trained model are frozen, and the only parameters that are fine-tuned are the adapters. It allows the model to converge faster as it is only necessary to train a few numbers of parameters (especially when compared with the total size of the model). Furthermore, it is possible to train adapters separately or simultaneously. The main disadvantage of this approach is the necessity of having a component for each task. If the number of tasks (in machine translation each language represents a task) increases, the number of parameters will increase too.

They were initially introduced for computer vision (He et al. [49]. Regarding Natural Language Processing, Houlsby et al. [50] proposed its use for language modelling. Later, Pfeiffer et al. [51] achieved strong results in multi-task and cross-lingual transfer learning.

In the field of multilingual neural machine translation, adapters were introduced by Bapna and Firat [26] to recover the performance that is usually lost in the high-resource settings when they are trained together with low-resource languages. Fine-tuning with adapters, allows the model to see larger portions of the data for the language-pairs that have adapters.

In this subsection, the different possible architectures of these layers are presented, followed by an explanation of its training process. Then, the current approaches in the field of multilingual neural machine translation are presented. Afterwards, this work is extended by introducing some changes that further improve its performance, and allow their use in a direct zero-shot scenario.

3.3.1 Adapters' Architecture

Houlsby et al. [50] were the first to propose the use of adapters for NLP tasks. The authors demonstrated that the placement and architecture of the adapters in the transformer is not a trivial problem. They experimented different architectures and placements and concluded that a two-layer feed-forward neural network with a bottleneck works well. They placed two of these adapters within each transformer layer (both in the encoder and decoder), one after the multi-head attention and one after the feed-forward

layer (Figure 3.1). During fine-tuning, they only needed to re-train the layer normalization layers and the adapters.

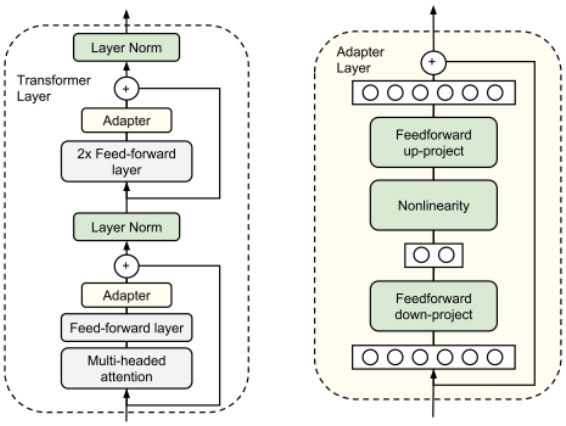

Figure 3.1: Simplified representation of the adapters proposed in [50]. Taken from [50].

More recently, Pfeiffer et al. [52] performed an exhaustive search over different Adapter architectures. The best performing architecture ended up being a single adapter after the Feed-Forward network with a residual connection connecting the output of the Feed-Forward with the output of the Adapter. The residual connection enables a pass-through and by doing this, it allows keeping at least the performance of the pre-trained model.

In the field of multilingual NMT, Bapna and Firat [26] achieved better results using only one adapter (after the feed-forward layer). However, they have introduced a layer normalization and recurrent connections. They introduced a layer normalization in each adapter to avoid retraining all the existing layer normalization layers, as was done by Houlsby et al. [50]. The residual connections are introduced to allow the module to represent a no-operation if necessary. The hidden dimension of the adapter is the hyperparameter that is fine-tuned. They argue that this strategy allows them to adjust the capacity of the adapter easily, depending on the task they want to perform, adjusting only one hyperparameter. In Figure 3.2, it is shown this adapter layer and its placement in the transformer layers, after the feed-forward network of each layer.

We decided to implement the same architecture as Bapna and Firat [26], a layer normalization followed by a feed-forward network and recurrent connections. Our work is different from Bapna and Firat [26] because we propose the use of adapters for all language-pairs (and not only for high-resource ones) and propose to condition the adapters only on one language instead of a language-pair. The implementation is available online. [1]

3.3.2 Adapters' Training Process

The injection and training of adapters is a two-step algorithm: firstly, it is necessary to train a fully shared model on all language-pairs, and then there is a fine-tuning using only adapters. The adapters can be

[1] Code available online at:https://github.com/JoaoMCAlves/Multilingual-Adapters

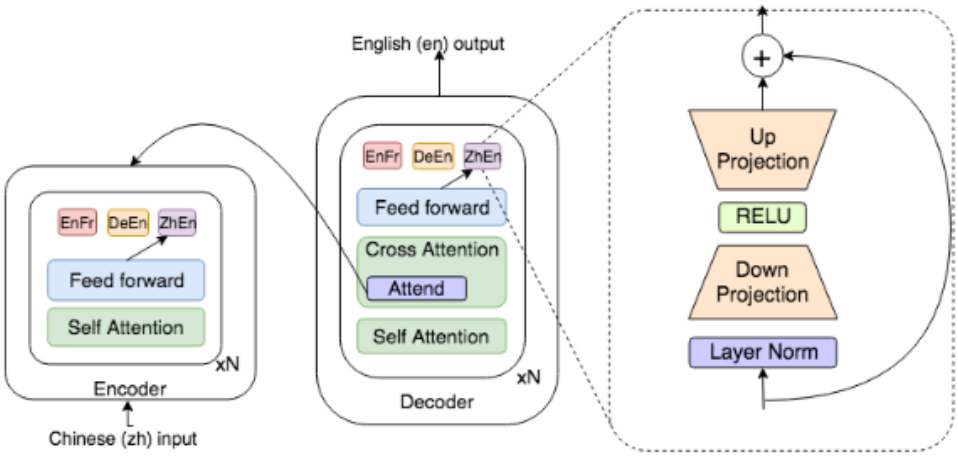

Figure 3.2: Adapters proposed by Bapna and Firat [26] and their placement in the Transformer's layers. Taken from Bapna and Firat [26].

updated sequentially on each task or they ca be trained simultaneously. Regarding the training data used to fine-tune the adapters, it is possible to use the same training data used for training the fully-shared system or it is possible to choose a different dataset.

Adapters are trained in the same way as full fine-tuning of the pre-trained model. The data is passed through all layers of the transformer. It is used in the same settings as the fully shared system but the learning rate schedule is reset. All the parameters are frozen, except the adapters.

3.3.3 Proposed Changes: Language-Specific Adapters

In the case of the many-to-many systems, if we want to have adapters for all languages, the naive approach would be injecting language-pair specific adapters (as was suggested in Bapna and Firat [26]). However, there are a few issues that arise: the first and obvious one is that we would need to have twice the number of adapters compared to the previous experiments ($N-1$ from any language to English and $N-1$ from English to any language).

Moreover, it would not allow us to use adapters to perform direct zero-shot translations. The work developed by Bapna and Firat [26] did not take into consideration the possibility of performing direct zero-shot translation. As the adapters would be conditioned on English either in the source or in the target, we would have a problem if we wanted to translate between non-English languages.

In an extreme case, we could have an adapter for each possible language combination. However, as we might not have parallel data for every language-pair, we would not be able to train all the adapters and it would imply having $O(N^2)$ adapters. Adapters have a small number of parameters, but if we have such a huge number of adapters, the number of parameters is going to increase quadratically with the number of languages.

To solve this problem we decided that it could be beneficial to have adapters conditioned only on one language, instead of a language-pair. We propose three different architectures based on the language

that conditions the choice of the adapters:

- source-specific adapters both in the encoder layers and in the decoder layers;
- target-specific adapters both in the encoder layers and in the decoder layers;
- source-specific adapters in the encoder layers and target-specific adapters in the decoder layers.

In the three different configurations, we are going to have the same number of adapters in each layer. The number of adapters per layer is going to be equal to the number of languages of our dataset, N.

In this chapter we have described a few different works that tried to condition components/mechanisms on the source or the target language. For example, Blackwood et al. [37] tried different kinds of attention: target-specific attention, source-specific attention and paired attention. They also added different tokens to their source sentences: source tokens, target tokens and pair tokens. In both situations, the best results were obtained when conditioning on the target language. Zhang et al. [34] conditioned some components of the transformer on the target language (layer normalization and linear transformation between the encoder and the decoder) and was able to achieve very good results. Due to these reasons, we are expecting to achieve the best results when conditioning all the adapters on the target side or when using a mixed approach, conditioning on the source side in the encoder and on the target side in the decoder.

Direct Zero-Shot Translation

Bapna and Firat [26] did not focus on direct zero-shot translation. The language-pair specific adapters were conditioned on English either on the source or on the target side. To translate between non-English languages, it would not be possible to take advantage of adapters. The language-specific adapters do not have this problem. As they are conditioned only on one language, it is always possible to use them regardless of architecture choice.

For example, if we want to translate from Portuguese to Spanish:

- if we choose to use source-specific adapters both in the encoder layers and in the decoder layers, the model is going to use the adapters conditioned on Portuguese both in the encoder and in the decoder;
- if we choose to use target-specific adapters both in the encoder layers and in the decoder layers, the model is going to use the adapters conditioned on Spanish both in the encoder and in the decoder;
- if we choose to use source-specific adapters in the encoder layers and target-specific adapters in the decoder layers, the model is going to use the adapters conditioned on Portuguese in the encoder and the adapters conditioned on Spanish in the decoder.

3.4 Summary

This chapter presented the existing approaches to the problem of multilingual neural machine translation. Multilingual NMT systems are appealing for some reasons. The first one is the number of models and consequently the number of parameters that they require. Another benefit is the fact that multilingual models can improve the performance on low and medium resource languages, especially when they are trained together with similar high-resource languages. Positive transfer learning from high-resource languages to low-resource ones helps to improve the results. In extreme cases, it is possible to translate between language-pairs that were never seen during training time (zero-shot translation). Multilingual systems can be the solution to the data scarcity problem, using languages with large-scale parallel data to improve the translation quality of those with less parallel data.

espite the potential benefits of multilingual approaches, they tend to underperform bilingual baselines, especially for high-resource languages (Johnson et al. [11], Arivazhagan et al. [9]). They tend to have considerably worse results when many languages are accommodated (Aharoni et al. [10]), in particular when languages are not related. Adding more languages may result in a bottleneck because the models have a limited capacity.

First, we described the different possible architectures: Universal Encoder-Decoder approaches and Language-specific Encoders and Decoders approaches. Afterwards, we explained the concept of transfer learning. In particular, we explored an extreme case of transfer learning: zero-shot translation. Zero-shot can be divided into two categories: direct zero-shot translation and pivot-based zero-shot translation. Regarding pivot-based zero-shot translation, we analysed the possibility of having different pivot languages.

Finally, we presented a strategy to improve the results of fully shared systems: the injection of adapters. The main advantage of this approach, when compared with others, is that it does not require full fine-tuning. adapters are residual layers introduced in the middle of a pre-trained model used to improve its performance. We propose the use of language-specific adapters instead of language-pair specific adapters. The main advantages of this approach are that it requires an inferior number of adapters and it allows the use of adapters for direct zero-shot translation. We will validate these models empirically in Chapter 4.

Chapter 4

Experimental Analysis

In this chapter, we describe our experiments, and present and analyse the results . First, we present the datasets used in our experiments. Then, we describe the baselines that are going to be compared against. After this, we are ready to present the results that we have obtained using multilingual NMT models. First, we present the results achieved using fully shared models (Section 4.5) in three different settings: many-to-one, one-to-many and many-to-many.

Afterwards, the results achieved with adapters are presented in Section 4.6. For this approach, two sets of experiments are presented. First, we introduce language-pair specific adapters on top of the many-to-one and one-to-many model. Then we introduce language-specific adapters on top of the many-to-many model.

Later we evaluate the translation between non-English languages (Section 4.7). As it was exposed in Chapter 3 there are two possible approaches: pivot-based zero-shot translation and direct zero-shot translation. We report the results for both approaches and then compare them with the results achieved with bilingual direct models.

Finally, we evaluate the impact of using different pivot languages. Pivoting through English is the common approach. Our goal is to evaluate the influence of factors like language relatedness in the final results. In Section 4.8 we explore the use of German and French as pivot languages.

4.1 Datasets

The data has a significant impact on the systems that we want to develop.

In our case, the main goal is to cover 24 EU Official Languages: English (en), Bulgarian (bg), Czech (cs), Danish (da), German (de), Greek (el), Spanish (es), Estonian (et), Finnish (fi), French (fr), Hungarian (hu), Italian (it), Latvian (lv), Lithuanian (lt), Dutch (nl), Polish (pl), Portuguese (pt), Romanian (ro), Slovak (sk), Slovenian (sl), Swedish (sv), Irish (ga), Maltese (mt) and Croatian (hr).

For the majority of them, we used the Europarl dataset (Iranzo-Sánchez et al. [38]) as the training, development and test corpus. However, three languages are not covered by Europarl: Irish, Croatian and Maltese, so it was necessary to find alternatives. For Irish we have used *DGT* corpus (Steinberger

et al. [53]) and for Croatian and Maltese the *TildeMODEL* corpus (Rozis and Skadinš [54]). As the domain is not exactly the same, the transfer learning process is harder.

Statistics for the corpora are mentioned in Table A.1.

It is important to take into consideration that even our lowest resource languages exceed the amount of data available in a majority of the previously studied datasets, especially if we compare with other works that were performed like Arivazhagan et al. [9] or Kudugunta et al. [35]. For example in Kudugunta et al. [35] parallel sentences per language-pair range between 10^4 to 10^9. In our case, the low-resource ones have around 400k sentence pairs, and the high-resource ones have around 2M sentence pairs. Given the amount of data, some techniques that were developed for extremely low-resource datasets may not be as effective in our case. Because of that, we are not going to use oversampling strategies such as sampling temperature (Artetxe and Schwenk [55]). In this thesis, we are going to consider low-resource languages the ones that have fewer than 1M sentence pairs. The others are considered high-resource languages.

In order to be able to compare the results achieved for direct zero-shot Translation and Pivot when using languages of Europarl, we have created a test set and a development set common to all languages that Europarl supports. Namely, we used the data from the first semester of 2009 to accomplish this. The development set has 4000 sentences and the test set has 4630 sentences. After doing this, we guaranteed that there were no duplicates in the train set.

4.2 Data Preprocessing and Vocabulary Construction

The preprocessing step and the construction of the vocabulary take an important role in the creation of every neural machine translation system. We tokenized and truecased all the sentences using scripts from the Moses toolkit (Koehn et al. [56]).

We used Byte Pair Encoding (BPE) (Sennrich et al. [24]) to segment sentences into subword symbols and to construct all our vocabularies. For the bilingual experiments, the size of the shared vocabulary was 10k tokens and for multilingual experiments, we used a shared vocabulary containing 32k tokens. We decided to use this size based on the work developed by Arivazhagan et al. [9].

4.3 Metrics

As we are dealing with a large number of languages, it is important to find clear ways to present the results. Rather than providing the BLEU score for each language-pair, we are going to compute some average BLEU scores:

- $BLEU_{20}$ - average BLEU over all 20 language-pairs covered by *Europarl*
- $BLEU_{23}$ - average BLEU over all 23 language-pairs
- $BLEU_{HR}$ - average BLEU over the high-resource language-pairs
- $BLEU_{LR}$ - average BLEU over the low and medium resource language-pairs

Furthermore, it is possible to find complete results in appendix A.

4.4 Bilingual Baselines

To compare the performance of our multilingual models, we trained bilingual baselines using the training dataset that we have described before. We performed all our experiments using the open-source *Fairseq* toolkit (Ott et al. [57]).

The experiments were performed with transformer base settings (Vaswani et al. [7]), containing around 75M parameters. For these experiments, we used transformers with 6 layers in both the encoder and the decoder, model dimension set to 512, hidden dimension size of 512 and 8 attention heads. For optimization, we use Stochastic Gradient Descent with Adam Optimizer (Kingma and Ba [58]), ($\beta_1 = 0.9$, $\beta_2 = 0.98$) with label smoothing of 0.1 and scheduled learning rate (warm-up step 16k). Furthermore, we use dropout of 0.3. BLEU scores are calculated on the checkpoint with the best validation BLEU score employing beam search with a beam size of 5.

We plot the BLEU scores for different language-pairs in Figure 4.1. From left to right, languages are arranged in increasing order of available training data. We plot the two main directions separately in different plots. When English is the target language, we represent it as Any$\rightarrow$English. The same logic was applied for the opposite directions (when English is the source language), where we have English$\rightarrow$Any. The results of the multilingual models will be presented exactly the same way.

From Figure 4.1, it is possible to conclude that the translation quality is slightly better for high-resource languages. Furthermore, the translation quality is superior when English is the target language. Furthermore, the results on high-resource languages tend do be better than the ones for low-resource ones, especially when translating from English.

4.5 Fully Shared Models

Using the dataset that we described before, we trained three fully shared multilingual models: 1) many-to-one model from 23 languages to English, 2) one-to-many model from English into 23 languages and 3) many-to-many model trained using all 46 translation directions (to and from English).

For all settings, we train a single Transformer Base with the same hyper-parameters settings as the bilingual models. The only difference is the use of a shared BPE vocabulary with 32k tokens. Moreover, we followed the approach of Johnson et al. [11] and added a target-language token at the end of each source sentence. In all cases, we report test results (in terms of BLEU score) for the checkpoint that performed best on the validation set.

Another fact that is important to consider is the difficulties to accommodate out-of-domain language-pairs. As we have referred before, Europarl does not cover three official European languages: Croatian, Irish and Maltese. The deterioration in these languages is much higher, so we decided not to represent them in the same graphs as the other languages.

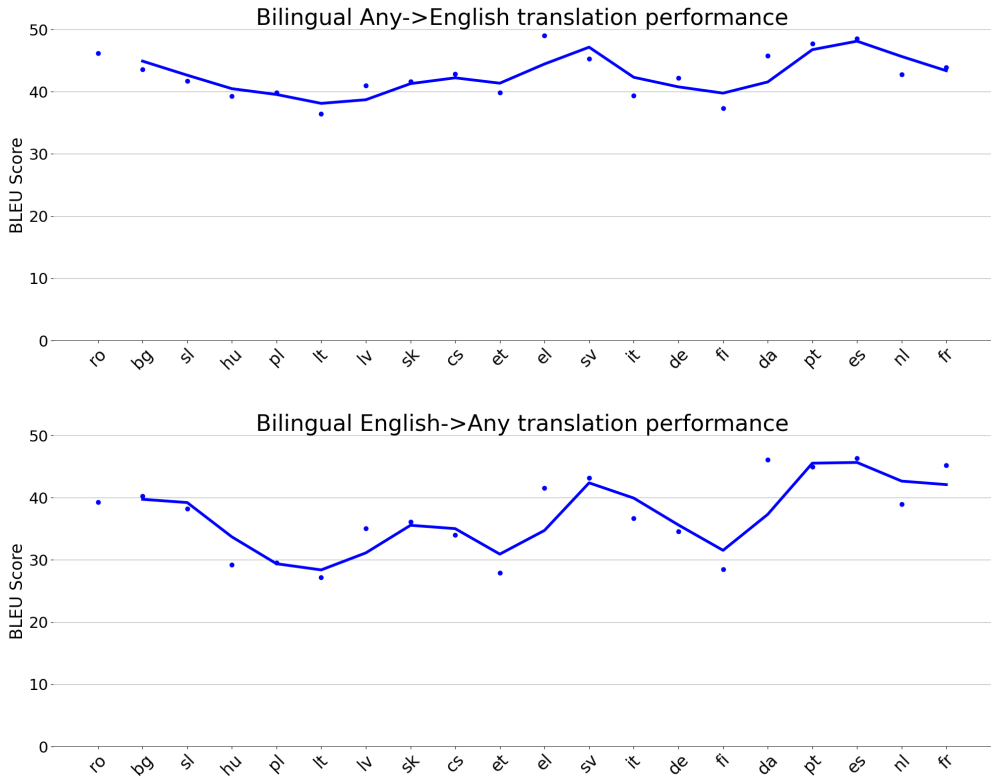

Figure 4.1: Quality of individual bilingual models on all 20 language-pairs covered by Europarl, measured in terms of BLEU score. Languages are arranged in increasing order of available data from left to right. The first plot reports BLEU for translating to English from any of the languages. The second one reports BLEU for translating from English to any of the other languages. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend.

4.5.1 Results

First, we trained a many-to-one and a one-to-many model. After having trained these models, we trained a many-to-many NMT model on the concatenation of the one-to-many dataset and the many-to-one dataset. By doing this we are able to perform direct zero-shot translation (translation between any language-pair) even if they do not have parallel data.

We plot the results when translating to and from English in Figure 4.2. Once again, we represented the two main translation directions in separate graphs. In the first plot, we represent the performance of the many-to-many model and the many-to-one when translating from all languages to English. In the second one, we represent the performance of the many-to-many and the one-to-many models when translating from English to other languages. Furthermore, we present some average metrics in Table 4.1. Complete results can be found in Tables A.2 and A.3.

The many-to-one shared model achieved worse results than the bilingual baselines if we consider all the languages of our dataset (-1,25 $BLEU_{23}$, 2-1, Table 4.1), but they were able to outperform the bilingual baselines if we consider only the languages that are covered by Europarl (+0,07 $BLEU_{20}$, 2-1, Table 4.1). It outperforms bilingual baselines when translation to English for low settings (+0,84 $BLEU_{LR}$ 2-1, Table 4.1). This phenomenon was already described in previous works like Aharoni et al. [10] and Neubig and Hu [59]. A multilingual system with shared weights promotes transfer learning from

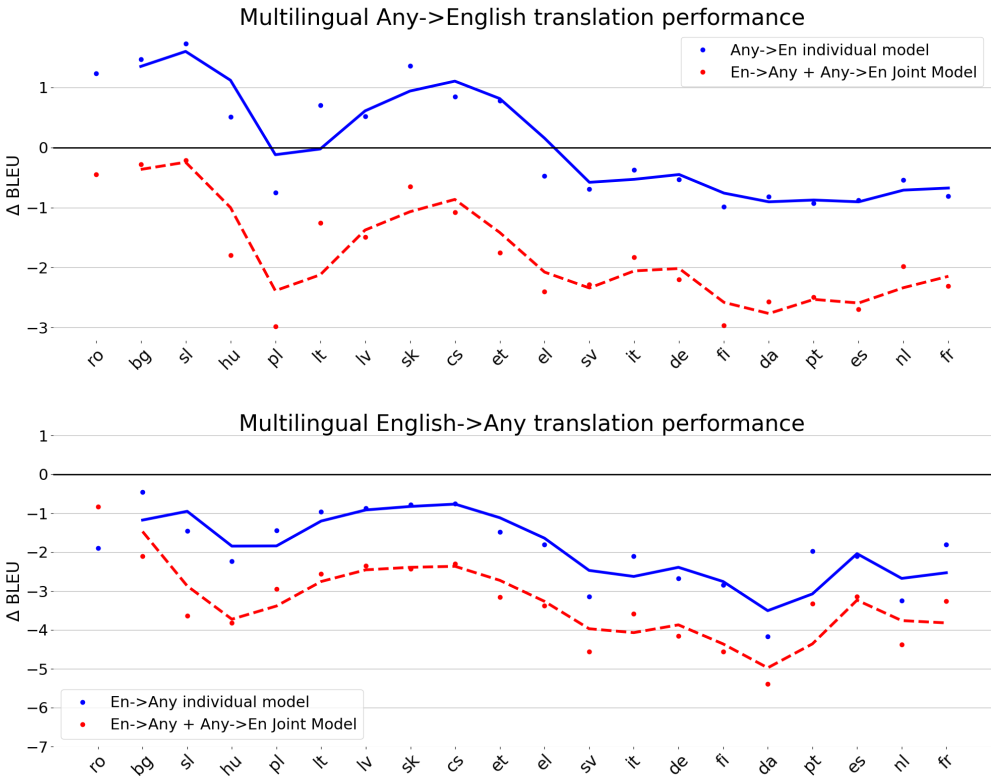

Figure 4.2: Quality of many-to-many, many-to-one and one-to-many models. Languages are arranged in increasing order of available data from left to right. The first plot reports BLEU for translating to English from any of the other 23 languages. The second one reports BLEU for translating from English to any of the other languages. Results are reported relative to those of bilingual baselines. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend. The colors corresponds to the following strategies : (i) Blue: Models trained in one translation direction (many-to-one and one-to-many respectively) (ii) Red: Model trained in both translation directions (to and from English). Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend.

Any→En	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
1.Bilingual	46.32	42.72	41.24	43.66
2.All → En	45.07	42.79	43.66	42.93
4.All → All	42.92	40.93	40.04	41.29
En→Any	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
1.Bilingual	36.90	37.14	33.67	40.51
3.En → All	34.25	35.03	32.02	37.84
4.All → All	33.04	33.85	31.05	36.47

Table 4.1: Average translation quality (BLEU score) of multilingual models trained. All → All reports the performance of the multilingual model trained on all translation directions, En → All reports the performance of the model trained on all language-pairs with English as the source and All → En reports the performance on the model trained on all language-pairs with English as the target.

high-resource languages to low-resource ones. On the other hand, the high-resource language-pairs have a decrease in their performance (-0,73 $BLEU_{HR}$ 2-1, Table 4.1). That might be explained by two different reasons:

- the different language-pairs are competing for capacity, and due to the limited model size, the

high-resource ones have difficulties to accommodate all the space that they need;

- the model converges before it trains on significant portions of the high-resource data.

Regarding the one-to-many setup, when we compare the results achieved with bilingual baselines, it is easy to realize that none of the language-pairs had an improvement in their performance. The deterioration in the performance is higher for high-resource languages (-2,67 $BLEU_{HR}$ 3-1, Table 4.1), while the performance on low-resource languages does not deteriorate that much (-1,64 $BLEU_{LR}$ 3-1, Table 4.1). As was described in Arivazhagan et al. [9], the many-to-one model can be seen as a multi-domain model, where each source language represents a specific domain. On the other hand, the one-to-many models can be seen as a multi-task model, with each target language representing a different task. This can help us understand why the process of transfer is far more evident in the case of many-to-one: transferring across multiple domains may be easier than transferring across multiple tasks. We can conclude that is harder to accommodate multiple languages in the target side.

Analysing the results of Figure 4.2, we notice both the many-to-one model and the one-to-many model achieve much better results than the many-to-many model. The deterioration is higher when translating to English (-2,15 $BLEU_{23}$ 4-2, Table 4.1) than when translating from English (-1,21 $BLEU_{23}$ 4-3, Table 4.1). The many-to-many model must accommodate twice as many translation directions with the same number of parameters. In this case, we have 46 translation directions instead of 23. Due to this, the many-to-many model suffers more relevant capacity issues.

Finally, it is important to take into account that we compared the different models under the same training conditions. We did not change the hyperparameters for each configuration to simplify experimentation. Probably if we change some hyperparameters, specially in the many-to-many settings, we would be able to achieve better results.

The obtained results were already expected according to the state-of-the-art (Arivazhagan et al. [9], Aharoni et al. [10]). In the majority of the cases, bilingual baselines outperformed the results achieved by multilingual models. In the next section, our goal is to improve the results and to close the gap between bilingual models and our multilingual models.

4.6 Adapters

After training fully shared models on all language-pairs, we inject and fine-tune adapters on top of them. It is important to take into consideration that our main goal is to close the gap to bilingual baselines, so we are always going to present the results in terms of the difference to the bilingual baselines.

First, we inject adapters on top of the many-to-one and one-to-many models. In this case, we inject language-pair specific adapters. Then we inject language-specific adapters on top of the many-to-many model.

For fine-tuning, we use the same hyper-parameter used during global pre-training but reset optimizer accumulator and restart from the first step.

4.6.1 Language-Pair Specific Adapters

In this subsection, the goal is to understand how language-pair specific adapters work. We have fine-tuned the many-to-one and the one-to-many with language-pair specific adapters on top. First, introduced a variable number of adapters, depending on the training data available for each language-pair, and evaluated the influence on the results. Then, we studied the effect of varying the dimension of the adapters.

Number of Adapters used

As it was explained in Section 3.3, we tested two different settings:

- injecting adapters only for high-resource languages;
- injecting adapters for all languages;

We tested it both on the many-to-one and the one-to-many models. We plot results in Figure 4.3. Further results can be found in Table 4.2. Complete results are in Appendix A.

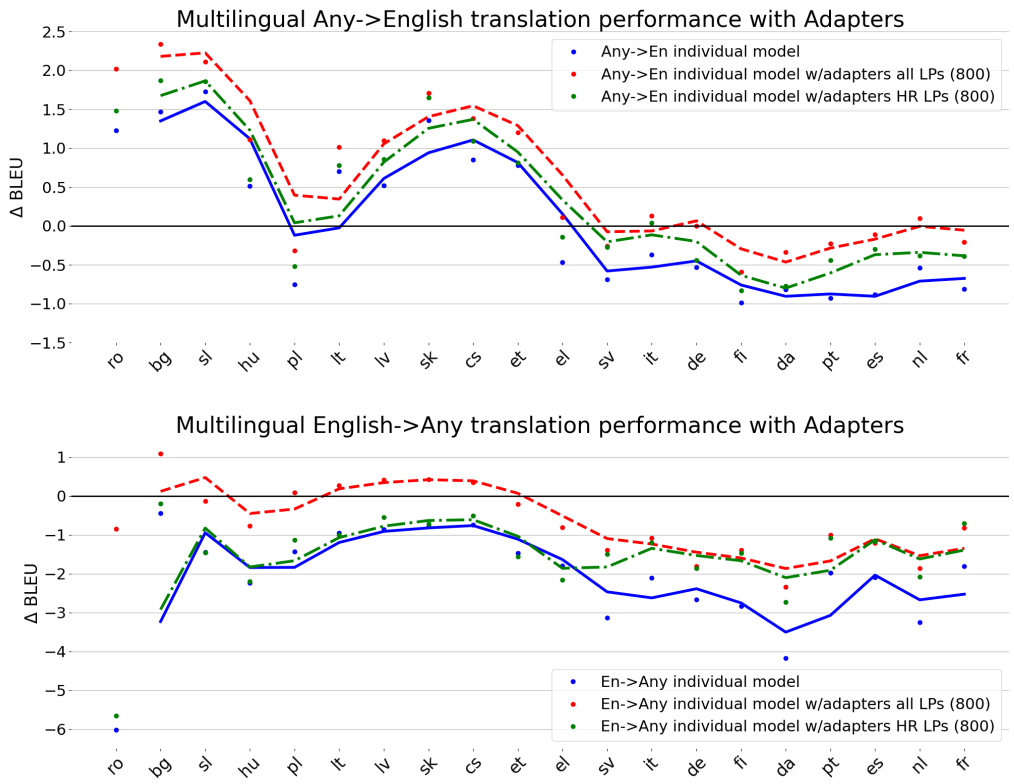

Figure 4.3: Effects of injecting adapters. Languages are arranged in increasing order of available data from left to right. The first plot reports BLEU for translating to English from any of the languages. The second one reports BLEU for translating from English to any of the other languages. Results are reported relative to those of bilingual baselines. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend. The colors corresponds to the following strategies : (i) Blue: Fully Shared Model (ii) Green: Fully Shared with adapters for high-resource languages (iii) Red: Fully Shared with adapters for all languages. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend.

Any → En	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
1.Fully Shared Model	45.07	42.79	42.08	42.93
2.Fully Shared Model w/Adapter HR	45.30	43.05	42.29	43.29
3.Fully Shared Model w/Adapter all	45.91	43.33	42.60	43.49
En → Any	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
4.Fully Shared Model	34.25	35.03	32.02	37.84
5.Fully Shared Model w/Adapter HR	34.75	35.59	32.17	39.98
6.Fully Shared Model w/Adapter all	35.83	36.49	33.73	39.07

Table 4.2: Average translation quality (BLEU score) of multilingual models trained. Fully shared model reports the performance of the multilingual model sharing all parameters (in the first case many-to-one, in the second one one-to-many), Fully Shared Model w/Adapter HR reports the performance of the shared models fine-tuned with adapters only for high-resource languages, Fully Shared Model w/Adapter all reports the performance of the shared models fine-tuned with adapters only for all languages.

First, we are going to analyse the first graph : **translating from any language to English**. Adapters clearly help to improve the performance of the model. There is an interesting phenomenon: even when we introduce adapters for high-resource languages only, the results for low-resource languages improve. This may be related to the fact that the embedding layer is not frozen. However, the best results are achieved when adapters were injected for all languages. This configuration achieved the best results for every single language.

Now, let's analyse the second plot: **translating from English to any language**. As we have seen before and unlike the many-to-one model, there is a degradation in the results for all languages pairs when a fully shared model is used. Analysing the graph, it is possible to conclude that when we inject adapters only for high-resource settings, there is a huge improvement for high-resource languages (+2,14 $BLEU_{HR}$, 5-4, Table 4.3). The introduction of language-pair specific for all language-pair was capable of achieving better results in general (+1,08 $BLEU_{23}$, 6-5, Table 4.3) but worst results if we consider only high-resource languages (-0,91 $BLEU_{HR}$, 6-5, Table 4.3). For some low-resource languages, it was even able to surpass bilingual baselines: Bulgarian, Latvian, Estonian, Slovakian and Czech.

It is clear that the performance improves by a huge margin after training the adapter's parameters. Furthermore, it is also true that injecting adapters for all language-pairs seems to be the better choice as it can achieve better results in general. From now on, we are going to inject adapters for all languages as the results are clearly better.

Adapters with different dimensions

The adapter architecture that we have chosen to implement has only one hyperparameter (Section 3.3): the hidden dimension of the adapter. To analyse the impact of this hyperparameter, we have introduced adapters with different hidden dimension on top of the many-to-one model. In Section 3.3, we have seen that it is better to inject adapters for all languages, so we did it here too.

We have tested two different dimensions (256 and 800) and we have compared the results with the ones obtained without adapters. Once again, we plot the scores relative to the bilingual baselines. We depict the results in Figure 4.4. Further results are presented in Table 4.3. Complete results are in

Appendix A.

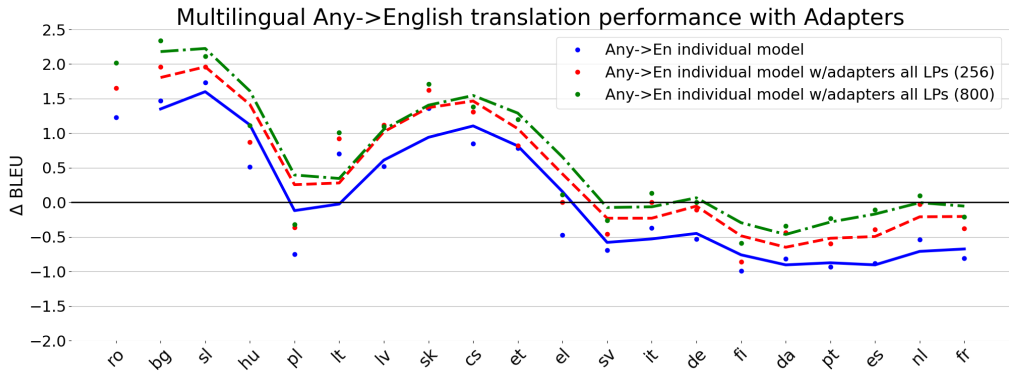

Figure 4.4: Effects of varying the hidden dimension of the adapter architecture . Languages are arranged in increasing order of available data from left to right. The plot reports BLEU for translating to English from any of the languages. Results are reported relative to those of bilingual baselines. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend. The colors corresponds to the following strategies : (i) Blue: Many-to-One fully shared model (ii) Red: Many-to-One w/Adapters for all languages (hidden dimension 256) (iii) Green: Many-to-One w/Adapters for all languages (hidden dimension 800). Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend.

Any → English	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
1.Fully Shared Model	45.07	42.79	42.08	42.93
2.Fully Shared Model w/Adapters (256)	45.56	43.15	42.42	43.30
3.Fully Shared Model w/Adapters (800)	45.91	43.33	42.60	43.49

Table 4.3: Average translation quality (BLEU score) of multilingual models trained. Fully Shared Model reports the performance of the many-to-one model that shares all parameters, Fully Shared Model w/Adapters (256) reports the performance of the many-to-one model with adapters on top with b=256, Fully Shared Model w/Adapters (800) reports the performance of the many-to-one model with adapters on top with b=800.

The results in Figure 4.4 indicate that the model performance improves when we increase the size of the adapters. If we look carefully, it is possible to see that the performance is better for every language-pair when we increase the dimension of the adapters (+0,35 $BLEU_{23}$, 3-2, Table 4.3). Both low-resource and high-resource take advantage of it. If we look at Table 4.3, the results are consistent, all the metrics have the highest results when is used the model with b=800.

It is possible to conclude that increasing the size of the adapters helps to improve the performance of the model. This shows the flexibility of the adapters, according to the target task it is possible to change the adapter size to achieve the desired results. However, it is important to take into consideration that adapters are designed to be small layers: if we increase too much their size, we are going to increase the model size too, and in extreme cases, adapters could be bigger than the single model. It is very important to control this trade-off.

4.6.2 Language-Specific Adapters

As described in Section 3.3, injecting language-pair specific adapters on top of the many-to-many model would not be a good choice. It would require a considerable number of adapters (and consequently a considerable number of parameters) and it would not allow us to take advantage of them to perform direct zero-shot translation. To solve this issue, we experimented three different adapter configurations:

1. Source-specific adapters both in the encoder and decoder layers;
2. Target-specific adapters both in the encoder and decoder layers;
3. Source-specific adapter in the encoder layers and Target-specific adapters in the decoder layers.

We plot the results in Figure 4.5. Further results are summarized in Table 4.4. Complete results are available in Appendix A.

Any→En	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
1.Many-to-Many	42.92	40.93	40.04	41.29
2.Many-to-Many w/Source Adapters	44.70	42.10	41.36	42.30
3.Many-to-Many w/Target Adapters	43.75	41.71	40.96	41.94
4.Many-to-Many w/Source+Target Adapters	44.42	42.14	41.41	42.32
En→Any	$BLEU_{23}$	$BLEU_{20}$	$BLEU_{LR}$	$BLEU_{HR}$
1.Many-to-Many	33.04	33.85	31.05	36.47
2.Many-to-Many w/Source Adapters	33.82	34.63	31.97	37.11
3.Many-to-Many w/Target Adapters	35.70	36.33	33.59	38.88
4.Many-to-Many w/Source+Target Adapters	35.28	36.00	33.33	38.51

Table 4.4: Average translation quality (BLEU score) of multilingual models trained. Many-to-Many reports the performance of the fully shared model trained on all translation directions, Many-to-Many w/Source Adapters reports the performance of the Fully Shared Model with adapters conditioned on the source language, Many-to-Many w/Target Adapters reports the performance of the Fully Shared Model with adapters conditioned on the target language, Many-to-Many w/Source+Target Adapters reports the performance of the Fully Shared Model with adapters conditioned on the source language in the encoder and conditioned on the target language in the decoder.

The injection of **target-specific adapters** show the largest benefit for **English→X** translation (+2,66 $BLEU_{23}$, 3-1, Table 4.4). Furthermore, the results were consistent, if we consider only the low-resource settings, the improvement was of +2,54 $BLEU_{LR}$ and if we consider the high-resource ones it was +2,41 $BLEU_{HR}$. For **X→English** translation, it shows a smaller benefit (+0,83 $BLEU_{23}$, 3-1, Table 4.4). Once again, the results were better for low-resource (+0,91 $BLEU_{LR}$, 3-1, Table 4.4) and high-resource settings (+0,64 $BLEU_{HR}$, 3-1, Table 4.4). As the adapters are conditioned on the target language, the capacity is mainly increased for English→X and consequently, the results are better in this translation direction.

On the other hand and as it was expected, **source-specific adapters** yield a larger benefit for **X→English** tasks (+1,78 $BLEU_{23}$, 2-1, Table 4.4) and a smaller benefit for **English→X** tasks (+0,78 $BLEU_{23}$, 2-1, Table 4.4). For both translation directions, the results have improved for low-resource and high-resource configurations.

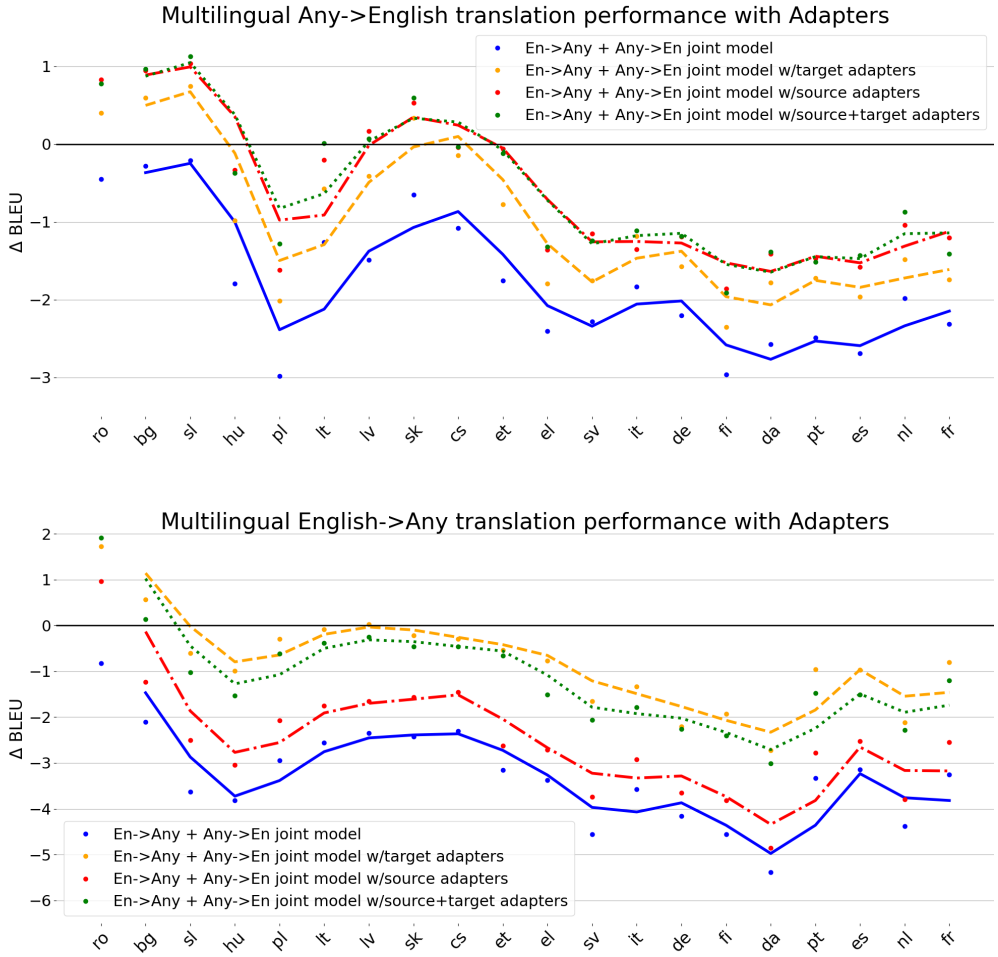

Figure 4.5: Effects of conditioning the adapters on different conditions. Languages are arranged in increasing order of available data from left to right. The first plot reports BLEU for translating to English from any of the other 23 languages. The second one reports BLEU for translating from English to any of the other languages. Results are reported relative to those of bilingual baselines. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend. The colors corresponds to the following strategies : (i) Blue: Fully Shared Model (ii) Orange: Fully Shared Model with adapters conditioned on the target language (iii) Red: Fully Shared Model with adapters conditioned on the source language (iii) Green: Fully Shared Model with adapters conditioned on the source language in the encoder and conditioned on the target language in the decoder. Performance on individual language-pairs is reported using dots and a trailing average is used to show the trend.

Although target and source-specific adapters were able to achieve good results (target adapters are better when translating from English to any language, and source adapters are better when translating from any language to English), a **combination of source-specific adapters in the encoder and target-specific adapters** in the decoder was able to achieve more balanced results, if we consider both translation directions. If we consider only English→X translation, it performs slightly worst than target adapters (-0,42 $BLEU_{23}$, 4-3, Table 4.4). But if we consider X→English translation, it was able to achieve better results than the exclusive use of source adapters (+0,67 $BLEU_{23}$, 4-2, Table 4.4).

To sum up, **the use of target adapters both in the encoder and in the decoder was the approach that achieved the best results when translating from English to any language, both for**

low-resource a high-resource settings. The combination of source adapters (in the encoder side) and target adapters (in the decoder side) was the setting that achieved the best results when translating from any language to English. Later, we are going to evaluate the performance of the three settings when used to generate direct zero-shot translations.

4.7 Translation between Non-English Languages

In the previous sections, we have presented the results when translating to and from English. In this section, our goal is to translate between any language-pair that we have in our dataset. We are going to present the results for two different approaches: **pivot-based zero-shot translation** and **direct zero-shot translation**. For pivot-based translation, we are going to compare the performance when we use only bilingual models for pivoting and when we use a many-to-one model followed by a one-to-many model. In the case of direct zero-shot translation, we are going to compare the performance of the fully shared models with the models with and without adapters on top. Finally, we trained a few direct bilingual models to compare the performance of these two methods with **direct translation**.

As it was described in Chapter 3, pivot-based translation approaches exploit the fact that the translation can be decomposed into a sequence of two steps: firstly translate from the source language to English and then translate from English to the target language. direct zero-shot translation does not require the intermediate step that we have for pivot translation. In this case, the goal is to directly translate from the source to the target language. This can only be performed using a multilingual many-to-many model.

4.7.1 Results on Pivot-Based Zero-Shot Translation

Regarding Pivot translation, we did two different experiments: **pivoting using the bilingual models** and **pivoting using a many-to-one model followed by a one-to-many model**. In the case of using the bilingual models, we need to have 46 models if we want to cover all the EU official languages (23 models from different languages to English and 23 models trained in the opposite direction). If we use the many-to-one and the one-to-many models, we only need to have 2 models. We use the one-to-many and the many-to-one models with language-pair specific adapters on top because this was the approach that achieved better results in the previous experiments (Section 4.6.1).

Although it is possible to perform zero-shot translation between any language-pair in our dataset, it is important to take into consideration that it is only possible to evaluate pivot translation between languages that are covered by *Europarl*. It is necessary to have a common test set between the languages to be able to evaluate the performance of the systems in this situation.

In Table 4.5 we compute the average values when each language is either the source (left side) or the target (right side). Tables with all results are in Appendix A.

The results achieved by the bilingual models were better than the ones obtained by the shared models. However, shared models achieved competitive results, being better for some languages com-

Source	Pivot Bilingual	Pivot Shared	**Target**	Pivot Bilingual	Pivot Shared
ro	29.59	**29.75**	ro	**28.45**	27.69
bg	28.54	**28.63**	bg	30.48	30.83
sl	28.01	**28.20**	sl	**28.41**	28.20
hu	**25.21**	24.94	hu	**22.86**	22.56
pl	26.81	**26.82**	pl	23.46	**23.56**
lt	**25.00**	24.91	lt	21.39	**21.58**
lv	**27.54**	27.34	lv	26.39	**26.59**
sk	28.20	**28.28**	sk	27.37	**27.44**
cs	28.35	**28.41**	cs	25.44	**25.58**
et	**27.49**	27.17	et	**20.86**	20.69
el	**30.14**	29.51	el	**29.4**	29.12
sv	**28.39**	27.85	sv	**31.19**	30.65
it	**26.36**	25.88	it	**29.11**	28.40
de	**27.80**	27.23	de	**25.99**	25.17
fi	**24.25**	23.33	fi	**22.20**	21.33
da	**28.57**	28.00	da	**33.26**	32.22
pt	**29.73**	29.07	pt	**32.81**	32.37
es	**29.65**	29.06	es	**33.92**	33.60
nl	**26.45**	25.85	nl	**30.44**	29.51
fr	**27.99**	27.30	fr	**35.18**	34.99
Average	27.71	27.38	**Average**	27.93	27.60
$BLEU_{LR}$	27.72	27.63	$BLEU_{LR}$	25.86	25.80
$BLEU_{HR}$	27.69	27.06	$BLEU_{HR}$	30.46	29.80

Table 4.5: Pivot-Based zero-shot translation results.

bination. If we compute an average of all the possible translation combinations, the degradation is only -0,33 BLEU points.

If we analyse only the left side of the table (source side) it is possible to conclude that, for high-resource languages, the results are always better when using bilingual models for pivoting and for some low-resource languages, it is possible to achieve better results when using the many-to-one model followed by the one-to-many model: Romanian (ro), Bulgarian (bg), Slovenian (sl), Polish (pl), Slovakian (sk) and Czech (cs).

If we take a look at the right-most column of Table 4.5, the conclusions are very similar, for high-resource settings bilingual models always achieve better results. For low-resource settings, Bulgarian (bg), Latvian (lv), Polish (pl), Slovakian (sk) and Czech (cs) achieved better results when using the shared models for pivoting,

Taking into consideration the number of models (and consequently the number of parameters) necessary to perform each of the strategies, we believe that using the shared models for pivoting is the best approach. Instead of using 46 models, we only need 2 and the results achieved are quite comparable (in some cases are even better) to the ones obtained with bilingual models.

4.7.2 Results on Direct Zero-Shot Translation

Many-to-Many models have the advantage over the single-task models of being able of translating between any pair of supported languages, even when parallel data is not available. Obviously, it is only able to perform translation between languages it has seen individually as a source or target languages during training time. As was described in the previous sections, transfer learning across different target

languages in a one-to-many model was already a hard process. It is not difficult to realize that in direct zero-shot translation that is going to be even harder. In this case, it is necessary to first transfer learning between different source languages and then between different target languages.

In this subsection, our goal is to perform direct zero-shot translation using the many-to-many models. As we have described before, we have trained three different settings using adapters on top of the many-to-many model. We have already concluded that they yield benefits when translating to and from English. But how do they impact the performance of direct zero-shot translation? We performed direct zero-shot translation using the many-to-many fully shared model and the models with adapters on top. The results are in Table 4.6. Once again, we compute the average values when each language is either the source (left side) or the target (right side). The complete results that we have obtained are in Appendix A.

	Many-to-Many				Many-to-Many		
Source	Fully Shared	Target Adapters	Source+Target Adapters	**Target**	Fully Shared	Target Adapters	Source+Target Adapters
ro	9.19	24.10	22.88	ro	18.71	25.32	25.81
bg	7.86	22.80	21.86	bg	19.52	27.67	27.12
sl	6.69	22.00	21.69	sl	14.22	24.59	22.80
hu	8.06	19.89	19.49	hu	12.56	19.95	18.36
pl	3.87	19.95	20.46	pl	16.62	20.56	20.11
lt	9.67	20.32	19.93	lt	9.68	18.56	17.28
lv	12.12	22.72	22.63	lv	15.96	23.69	23.05
sk	7.19	22.86	22.02	sk	16.53	23.29	23.26
cs	7.54	23.29	23.38	cs	13.38	19.77	20.46
et	9.99	22.15	21.74	et	12.75	18.01	17.88
el	17.12	25.65	24.83	el	15.22	25.80	25.07
sv	21.85	25.06	24.73	sv	13.01	24.02	25.04
it	15.87	22.87	22.68	it	10.03	23.96	21.67
de	18.66	24.38	23.72	de	9.51	20.86	20.66
fi	11.29	19.19	18.99	fi	9.79	17.17	16.50
da	21.63	25.23	24.80	da	9.29	25.76	25.66
pt	19.81	25.87	25.81	pt	12.13	25.32	25.68
es	18.26	25.66	25.24	es	14.54	27.83	28.34
nl	21.36	23.64	23.24	nl	10.53	24.25	24.41
fr	19.99	24.57	24.25	fr	17.01	29.80	29.11
Average	13.40	23.11	25.24	Average	13.55	23.31	22.91
$BLEU_{LR}$	9.03	22.34	21.90	$BLEU_{LR}$	15.01	22.47	21.93
$BLEU_{HR}$	18.75	24.05	23.72	$BLEU_{HR}$	11.76	24.33	24.12

Table 4.6: Direct zero-shot translation results.

If we use the fully shared model to generate direct zero-shot translations, the results that we obtain are poor, especially if we compare with the ones that we obtained using pivot-based techniques (Table 4.5).

Regarding the use of source-specific adapters, the results that we obtained are also poor. We can say that the translations generated are completely useless. If we manually analyse some of them it is easy to realise that almost everything is in English. Conditioning the translation on the source language clearly is not the best approach to achieve good results for direct zero-shot translation. As the results were so poor (around 3-4 BLEU points), we decided not to present them in Table 4.6.

Using only target adapters improve a lot the results achieved in direct zero-shot translation. Our results show that there an improvement of +9,71 BLEU points in average when compared with the Fully

shared model. If we analyse Table 4.6, we can conclude that the improvement is higher for low-resource language pairs ($BLEU_{LR}$). Later, we are going to analyse a few examples manually to try to understand how the target-specific adapters impacted the translations.

Finally, the combination of source adapters (in the encoder) and target adapters (in the decoder) was able to achieve good results too, but not as good as the ones obtained with only target-specific adapters. For some language-pairs, it was the approach that achieved the best results, but in general, we can say that the use of only target-specific adapters yields a larger benefit.

Qualitative Analysis

We have selected a few examples that help us show the problems that each model faces and the improvements that they give us. We present in Table 4.7 examples of the source sentence (1), the reference for translation (2), the translations generated by the fully shared model (3) and the translations generated by the model with target adapters (4). We present the results with target adapters because it was the strategy that achieved better results in general.

Example 1: fr-pt	
1. Source	nous ne pouvons légiférer dans lapos; abstrait !
2. Reference	não podemos criar leis no abstracto !
3. Zero-Shot - Fully Shared	não podemos legislar no abstract !
4. Zero-Shot - w/Target Adapters	não podemos legislar no abstracto !
Example 2: ro-pt	
1. Source	este o dilemă pe care trebuie să o soluţionăm împreună .
2. Reference	é um quebra-cabeças que devemos resolver em conjunto .
3. Zero-Shot - Fully Shared	it is a dilemma que we need to resolve together.
4. Zero-Shot - w/Target Adapters	é um dilema que temos de resolver juntos .
Example 3: sl-es	
1. Source	Gospa predsednica , rada bi se dotaknila sorodne zadeve .
2. Reference	señora Presidenta , me gustaría tratar un tema que está relacionado .
3. Zero-Shot - Fully Shared	Madam President , I would like to touch on related matters .
4. Zero-Shot - w/Target Adapters	señora Presidenta , me gustaría referirme a related matters .
Example 4: sl-es	
1. Source	poleg tega to nima nič opraviti z Evropsko unijo .
2. Reference	además , esto no tiene nada que ver con la Unión Europea .
3. Zero-Shot - Fully Shared	furthermore , this has nothing to do with the European Union .
4. Zero-Shot - w/Target Adapters	además , esto no tiene nada que ver con la Unión Europea .

Table 4.7: Examples of direct zero-shot translations generated by multilingual models.

If we analyse manually some of the translations generated by the fully shared model, it is possible to identify the off-target translation issue when we use the fully shared model. It corresponds to translate into a wrong target language. As we are using an English-centric dataset, the model, during training time, it is exposed to more English sentences (both on the source and target side). As a result of that, it is very common to find English words in the translations. Some times, it is even possible to find translation examples that are completely in English (examples 3 and 4). When we introduce the target-specific adapters, there is a clear improvement on the off-target translation issue, which may explain the increased translation performance.

These examples help us understand the importance of target adapters in direct zero-shot transla-

tions. When we use the fully shared many-to-many model without any modification it is possible to find many instances of off-target translations (examples 1 and 2) and in some cases sentences that have only English words (examples 3 and 4). After injecting the target adapters this problem is solved in the majority of the cases.

4.7.3 Comparison between Direct Zero-Shot Translation, Pivot-Based Zero-Shot Translation and Direct Translation

In the previous sections, we have analysed pivot-based zero-shot translation and direct zero-shot translation. In this subsection, we have trained a few direct bilingual models to be able to compare the results with the ones that we have obtained for pivot-based zero-shot translations and direct zero-shot translations.

The first thing that we need to take into consideration is the number of models that we need to perform translation between any EU official language in each of the approaches. If we want to cover translation between all 23 language-pairs we would need:

- **552 models** - if we use bilingual direct models;
- **46 models** - if we perform pivot translation using bilingual model, 23 that translate from any language to English and 23 that translate from English to any language;
- **2 models** - if we perform pivot translation with shared models, one model that translates from any language to English (many-to-one) followed by a model that translates from English to any language (one-to-many);
- **1 model** - if we perform direct zero-shot translation.

Moreover, pivot-based translation requires twice the translation time when compared with bilingual direct models or direct zero-shot translation.

The results are in Table 4.8. Regarding direct zero-shot translation, these results were obtained with the model with target-specific adapters on top.

Looking at the results in Table 4.8 it is possible to draw some conclusions. Comparing the results of pivot translation experiments towards the direct translation results, we can see that in general the pivot translation performs worse than the direct translation approach (-0.13 BLEU points on average). However, the pivot results are very competitive, and in some cases are even better.

It is also important to try to understand in which cases the direct bilingual models perform better. If we analyse it, we can see that it mainly happens when we are dealing with language-pairs from Romance family: it-fr, pt-fr, es-fr, fr-it, fr-pt and fr-es.

Regarding direct zero-shot translation performance, it is still behind the performance of pivoting through a common language, but the results are promising. The injection of target-specific adapters substantially narrows the performance gap with bilingual models as we have seen before.

To sum up, it is difficult to say which is the best approach. Bilingual direct models are still the ones that can achieve better results for most of the language combinations. However pivot-based translation

	Direct Bilingual	Pivot-Based Bilingual	Pivot-Based Shared Models	Direct Zero-Shot w/Target Adapters
de-fr	34.87	**35.06**	34.51	30.56
it-fr	**35.91**	34.83	34.54	30.84
pt-fr	**39.45**	38.63	37.81	34.28
es-fr	**39.04**	38.18	37.74	34.03
fr-de	25.89	**26.26**	25.00	22.39
fr-it	**31.41**	30.89	29.69	27.18
fr-pt	**35.38**	34.67	33.55	29.57
fr-es	**35.72**	35.53	34.60	31.70
da-de	**27.04**	**27.04**	25.96	24.31
nl-de	**27.22**	27.05	26.04	24.64
pl-de	24.70	**24.75**	24.26	17.13
sv-de	**27.01**	26.82	25.74	23.62
de-da	32.77	**33.29**	32.01	28.27
de-nl	**31.26**	31.25	29.84	27.38
de-pl	22.59	**23.21**	22.92	20.76
de-sv	30.46	**31.27**	30.53	26.86
Average	**31.30**	31.17	30.30	27.10

Table 4.8: Average translation quality (BLEU score) of different approaches for translating between non-English languages.

is able of achieving very competitive results with a lower number of models. Regarding direct zero-shot translation, it is not able to achieve comparable results yet, adapters clearly helped many-to-many models to boost their performance, but they were not able to completely close the gap. The choice is a trade-off between the number of models and the desired level of quality.

4.8 Use of Different Pivot Languages

To perform translation between languages that do not have parallel data we have tried two different techniques: direct zero-shot translation and pivot-based translation. As we were using an English-centric dataset, the pivot language that we have used so far was English.

Later we have asked ourselves if we could improve the performance if we use a different pivot language. English is the usual choice, but we want to check if the usage of a language that belongs to the same language family helps to improve the translation performance in the case of pivot translation.

We have identified two languages with potential to be used as pivot languages: **German** (de) and **French** (fr). These languages were chosen because they present a big volume of data and are representative of different language families. After having chosen these two languages, we have selected the languages that could be helped with this strategy:

- In the case of **French** we have chosen: **Portuguese** (pt), **Spanish** (es), **Italian** (it) and **German** (de);
- In the case of **German** we have chosen: **Polish** (pl), **Dutch** (nl), **Swedish** (sv), **French** (fr) and **Danish** (da).

We trained dedicated bilingual model for each language to and from English, and to and from the desired pivot language (French or German depending on the previous list). To train these models, we used the *Europarl* dataset. For these experiments, we used Transformer Base with 6 layers in both the encoder and the decoder, model dimension set to 512, hidden dimension size of 512 and 8 attention heads. For optimization, we use Stochastic Gradient Descent with Adam Optimizer (Kingma and Ba [58]), ($\beta_1 = 0.9$, $\beta_2 = 0.98$) with label smoothing of 0.1 and scheduled learning rate (warm-up step 16k). We use dropout of 0.3. BLEU scores are calculated on the checkpoint with the best validation BLEU score employing beam search with a beam size of 5.

After having trained the bilingual dedicated models, we used them for pivoting. We compare the translation quality of models of pivot translation in Tables 4.9 (for French) and 4.10 (for German).

	de		it		pt		es	
	English	French	English	French	English	French	English	French
de	-	-	28.75	27.40	32.73	30.34	33.63	31.45
it	24.47	22.92	-	-	32.80	32.01	33.43	32.46
pt	27.75	25.93	32.31	31.61	-	-	38.12	36.32
es	27.53	25.87	31.87	31.09	36.98	35.26	-	-

Table 4.9: Average translation quality (BLEU) of pivot translations using different pivot languages: English and French.Rows indicate source language, columns indicate target language.

	da		nl		fr		pl		sv	
	English	German	English	German	English	German	English	German	English	German
da	-	-	32.11	29.93	36.14	33.08	23.70	21.48	34.77	30.90
nl	34.50	31.08	-	-	35.64	33.32	23.45	21.04	32.22	28.48
fr	33.99	29.95	31.10	29.12	-	-	23.25	20.92	31.78	27.73
pl	30.89	28.24	29.00	27.22	33.57	30.74	-	-	29.30	26.67
sv	36.90	32.45	31.31	29.12	35.93	32.57	23.77	21.47	-	-

Table 4.10: Average translation quality (BLEU) of pivot translations using different pivot languages: English and German.Rows indicate source language, columns indicate target language.

Somewhat surprisingly, we see that it is clear that English is the best pivot language. The reasons behind it may be:

- the bilingual models to and from French have a worse performance (in terms of BLEU score) than the bilingual models to and from English. As the models that are used for pivoting are worse, consequently the pivot results are worse too. In the case of German, degradation is even higher;

- French and especially German are morphologically rich languages. The degradation in the case of using German as pivot language (-3,35 BLEU points on average when compared with results obtained using English) is much higher than in the case of French (-1,48 BLEU points).

Probably, if we have larger datasets, some of the problems might be solved. As we are dealing with morphologically rich languages, the size of the dataset has a huge impact on the final results.

4.9 Summary

Throughout this chapter, we have explored different ways of improving multilingual neural machine translation systems. The focus was on translating between all the official European languages.

As our main goal was to close the gap to bilingual dedicated models, first we have trained dedicated bilingual baselines for each language-pair available in our dataset. Then, we took the easiest way and trained fully shared models on three different configurations: many-to-one, one-to-many and many-to-many. After having concluded that fully shared models are still behind bilingual direct models (especially for one-to-many and many-to-many configurations), we explored the use of adapters on top of them to improve the results.

Adapters clearly help to close the gap between bilingual models (our baselines) and multilingual models. Probably if we increase even more the size of the adapters, the results would be even better. However, it is impossible to forget that the aim of this architecture is to introduce only a small number of parameters. If we increase too much the hidden dimension, we can possible loose the main advantage that adapters have.

Regarding translation between non-English languages, we tried two different approaches: pivot-based zero-shot translation and direct zero-shot translation. pivot-based was the one that achieved the best results (in some cases it was even possible to surpass bilingual baselines). Nevertheless, the introduction of target-specific adapters on top of a many-to-many fully shared system clearly helped us to close the gap to bilingual baselines.

Finally, we performed some experiments using different pivot languages. Contrary to what we expected, English achieved the best results for all the pivot translation that we have performed.

Chapter 5

Conclusions

In this chapter, the main achievements of this thesis are presented and we suggest some possible directions for future work.

5.1 Achievements

The goal of this thesis was to develop multilingual neural machine translation systems that cover all 24 European official languages. We show that it is possible to train multilingual models in large scale settings and that they can improve performance over bilingual baselines, especially for low-resource language-pairs.

We started by training fully shared models on three different configurations (many-to-one, one-to-many and many-to-many). We compared the results of these systems with bilingual baselines. Although the results were competitive, bilingual baselines were able to achieve better results in almost every translation direction.

In order to improve the performance of fully-shared multilingual NMT systems, we followed Bapna and Firat [26] and introduced adapters on top of the three fully shared models. Adapters enable a multilingual model to adapt to multiple target tasks without forgetting the original parameters of the model. In the case of the many-to-one and one-to-many models, we used language-pair specific adapters and evaluated the influence of varying the size of the adapters and the number of adapters. We concluded that the use of larger adapters yield improvements in terms of the BLEU score achieved. Furthermore, we concluded that the best choice was to use adapters for all language-pairs. In the case of the many-to-many model, we explored different kind of adapters. We tried to use adapters compatible with the idea of direct zero-shot translation. So, we injected three different adapters on top of the fully shared many-to-many systems: adapters conditioned on the source language both in the encoder and in the decoder, adapters conditioned on the target language both in the encoder and decoder and adapters conditioned on the source language in the encoder and on the target language in the decoder. The exclusive use of source adapters was the approach that achieved worst results in all scenarios. When translating from English to other languages, the exclusive use of target adapters was the best choice.

When translating from other languages to English, the hybrid approach was the one that had the best performance. In terms of direct zero-shot translation (when translating between non-English languages), target adapters, both in the encoder and decoder, were the ones that achieved the best results.

Regarding pivot-based zero-shot translation, we explored the use of different pivot languages. English is the usual choice as pivot language but we explored the use of French and German to translate between Romance and Germanic languages, respectively. We were expecting to have better results when using French or German for pivoting between languages from the same family. However, that did not occur. The best results were obtained when using English as the pivot language.

5.2 Future Work

In this work, we have injected adapters in all layers of fully-shared transformer systems. It would be interesting to test the use of these tiny residual layers in only some layer, i.e., only in the decoder side. Possibly, we could conclude that they have an higher influence in certain layers and increase their size in these layers or remove the adapters from the layers where they do not have impact.

Recent works (Pfeiffer et al. [52]) proposed to combine different adapters (*AdapterFusion*) for different Natural Language Processing tasks such as sentiment analysis, commonsense reasoning, paraphrase detection, and recognizing entailment. They use a mechanism very similar to the transformer's attention. The output of the feed-forward layer is their query vector (Q) and the output of each adapter is used both as the value (V) and key (K) vectors. One interesting direction direction to be explored is the use of this technique for multilingual neural machine translation. Sharing knowledge from multiple adapters could possible improve, even more, the results obtained due to transfer learning.

We believe our findings may inspire future research on multilingual NMT.

Bibliography

[1] A. Toral. Post-editese: an Exacerbated Translationese. *arXiv*, 2019.

[2] S. Läubli, C. Amrhein, P. Düggelin, B. Gonzalez, A. Zwahlen, and M. Volk. Post-editing productivity with neural machine translation: An empirical assessment of speed and quality in the banking and finance domain. *arXiv*, 2019.

[3] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993. URL `https://www.aclweb.org/anthology/J93-2003`.

[4] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana and Chicago, 1949.

[5] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 4(January):3104–3112, 2014. ISSN 10495258.

[6] D. Bahdanau, K. H. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–15, 2015.

[7] A. Vaswani, G. Brain, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention Is All You Need. *Advances in neural information processing systems*, (Nips): 5998–6008, 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

[8] D. Dong, H. Wu, W. He, D. Yu, and H. Wang. Multi-Task learning for multiple language translation. *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, 1:1723–1732, 2015. doi: 10.3115/v1/p15-1166.

[9] N. Arivazhagan, A. Bapna, O. Firat, D. Lepikhin, M. Johnson, M. Krikun, M. X. Chen, Y. Cao, G. Foster, C. Cherry, W. Macherey, Z. Chen, and Y. Wu. Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges. 2019. URL `http://arxiv.org/abs/1907.05019`.

[10] R. Aharoni, M. Johnson, and O. Firat. Massively Multilingual Neural Machine Translation. pages 3874–3884, 2019. doi: 10.18653/v1/n19-1388.

[11] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5: 339–351, 2017. ISSN 2307-387X. doi: 10.1162/tacl_a_00065.

[12] O. Firat. Zero-Resource Neural Machine Translation with. pages 268–277, 2016.

[13] C. G. Drury and J. Ma. Do language barriers result in aviation maintenance errors? *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 47(1):46–50, 2003. doi: 10.1177/154193120304700110. URL https://doi.org/10.1177/154193120304700110.

[14] C. Drury and C. V. Marin. Language error in aviation maintenance. 2005.

[15] K. Cho, B. van Merrienboer, H. Schwenk, and Y. Bengio. Learning Phrase Representation using RNN Encoder-Decoder. *Proceedings of the Empherical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[16] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. doi: 10.1109/5.58337.

[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

[18] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

[19] D. E. Rumelhart and J. L. McClelland. *Learning Internal Representations by Error Propagation*, pages 318–362. 1987.

[20] S. M. Lakew, M. Cettolo, and M. Federico. A Comparison of Transformer and Recurrent Neural Networks on Multilingual Neural Machine Translation. pages 641–652, 2018. URL http://arxiv.org/abs/1806.06957.

[21] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[22] W. Wang, J.-T. Peter, H. Rosendahl, and H. Ney. CharacTer: Translation edit rate on character level. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 505–510, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2342. URL https://www.aclweb.org/anthology/W16-2342.

[23] W. Ling, I. Trancoso, C. Dyer, and A. W. Black. Character-based neural machine translation, 2015.

[24] R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units, 2016.

[25] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

[26] A. Bapna and O. Firat. Simple, Scalable Adaptation for Neural Machine Translation. pages 1921–1931, 2019. doi: 10.18653/v1/n19-1191.

[27] S. M. Lakew, A. Erofeeva, M. Negri, M. Federico, and M. Turchi. Transfer Learning in Multilingual Neural Machine Translation with Dynamic Vocabulary. 2018. URL http://arxiv.org/abs/1811.01137.

[28] M. T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser. Multi-task sequence to sequence learning. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, (c):1–10, 2016.

[29] O. Firat, K. Cho, and Y. Bengio. Multi-way, multilingual neural machine translation with a shared attention mechanism. *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, pages 866–875, 2016. doi: 10.18653/v1/n16-1101.

[30] R. Vázquez, A. Raganato, J. Tiedemann, and M. Creutz. Multilingual NMT with a Language-Independent Attention Bridge. pages 33–39, 2019. doi: 10.18653/v1/w19-4305.

[31] B. Zoph, D. Yuret, J. May, and K. Knight. Transfer learning for low-resource neural machine translation. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 1568–1575, 2016. doi: 10.18653/v1/d16-1163.

[32] Y. Lu, P. Keung, F. Ladhak, V. Bhardwaj, S. Zhang, and J. Sun. A neural interlingua for multilingual machine translation. pages 84–92, 2019. doi: 10.18653/v1/w18-6309.

[33] C. Escolano, M. R. Costa-jussà, J. A. R. Fonollosa, and M. Artetxe. Multilingual Machine Translation: Closing the Gap between Shared and Language-specific Encoder-Decoders. 2020. URL http://arxiv.org/abs/2004.06575.

[34] B. Zhang, P. Williams, I. Titov, and R. Sennrich. Improving Massively Multilingual Neural Machine Translation and Zero-Shot Translation. 2020. URL http://arxiv.org/abs/2004.11867.

[35] S. Kudugunta, A. Bapna, I. Caswell, and O. Firat. Investigating Multilingual NMT Representations at Scale. pages 1565–1575, 2019. doi: 10.18653/v1/d19-1167.

[36] T.-L. Ha, J. Niehues, and A. Waibel. Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder. 2016. URL http://arxiv.org/abs/1611.04798.

[37] G. Blackwood, M. Ballesteros, and T. Ward. Multilingual Neural Machine Translation with Task-Specific Attention. pages 3112–3122, 2018. URL http://arxiv.org/abs/1806.03280.

[38] J. Iranzo-Sánchez, J. A. Silvestre-Cerdà, J. Jorge, N. Roselló, A. Giménez, A. Sanchis, J. Civera, and A. Juan. Europarl-ST: A Multilingual Corpus For Speech Translation Of Parliamentary Debates. 2019. URL http://arxiv.org/abs/1911.03167.

[39] E. A. Platanios, M. Sachan, G. Neubig, and T. M. Mitchell. Contextual parameter generation for universal neural machine translation. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, (2016):425–435, 2020. doi: 10.18653/v1/d18-1039.

[40] M. T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015. doi: 10.18653/v1/d15-1166.

[41] M. Post, B. MacCartney, M. Galley, C. D. Manning, D. P. Kingma, J. L. Ba, H. Schwenk, J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. N. Dauphin, M. T. Luong, H. Pham, C. D. Manning, J. Iranzo-Sánchez, J. A. Silvestre-Cerdà, J. Jorge, N. Roselló, A. Giménez, A. Sanchis, J. Civera, A. Juan, M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, M. Auli, G. Wentzel, Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Ł. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, J. Dean, T. Pires, E. Schlinger, D. Garrette, H. Wang, D. D. Yu, K. Sun, J. Chen, D. D. Yu, S. Rönnqvist, J. Kanerva, T. Salakoski, F. Ginter, J. L. Ba, J. R. Kiros, G. E. Hinton, K. Cho, B. van Merrienboer, H. Schwenk, Y. Bengio, W. Chan, N. Jaitly, Q. V. Le, O. Vinyals, P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, G. Neubig, R. Sennrich, B. Haddow, A. Birch, K. Cho, B. van Merrienboer, D. Bahdanau, Y. Bengio, N. Kalchbrenner, P. Blunsom, N. Kitaev, Ł. Kaiser, A. Levskaya, T. Kudo, J. Richardson, I. Sutskever, O. Vinyals, Q. V. Le, F. Casacuberta Nolla, Á. Peris Abril, M. Popel, and O. Bojar. Learning Phrase Representation using RNN Encoder-Decoder. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 3(1):1–12, 2015. ISSN 1578-7559. doi: 10.1002/andp.19223712302. URL http://arxiv.org/abs/2001.04451http://arxiv.org/abs/1703.01619http://dl.acm.org/citation.cfm?id=1557821http://arxiv.org/abs/1508.01211http://arxiv.org/abs/1607.06450http://arxiv.org/abs/1910.03806http://arxiv.org/abs/1609.08144http://arxiv.org/abs/.

[42] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, Oct. 2010. ISSN 1041-4347. doi: 10.1109/TKDE.2009.191. URL https://doi.org/10.1109/TKDE.2009.191.

[43] S. Ruder. *Neural Transfer Learning for Natural Language Processing*. PhD thesis, National University of Ireland, Galway, 2019.

[44] J. Kunze, L. Kirsch, I. Kurenkov, A. Krug, J. Johannsmeier, and S. Stober. Transfer learning for speech recognition on a budget. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 168–177, Vancouver, Canada, Aug. 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2620. URL https://www.aclweb.org/anthology/W17-2620.

[45] Z. Lu, Y. Zhu, S. J. Pan, E. W. Xiang, Y. Wang, and Q. Yang. Source free transfer learning for text classification. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, page 122–128. AAAI Press, 2014.

[46] X. Dong and G. de Melo. A helping hand: Transfer learning for deep sentiment analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2524–2534, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1235. URL https://www.aclweb.org/anthology/P18-1235.

[47] Y. Chen, Y. Liu, Y. Cheng, and V. O. Li. A teacher-student framework for zero-resource neural machine translation. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:1925–1935, 2017. doi: 10.18653/v1/P17-1176.

[48] A. Currey and K. Heafield. Zero-Resource Neural Machine Translation with Monolingual Pivot Data. (Wngt):99–107, 2019. doi: 10.18653/v1/d19-5610.

[49] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[50] N. Houlsby, A. Giurgiu, S. Jastrzçbski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:4944–4953, 2019.

[51] J. Pfeiffer, A. Rücklé, C. Poth, A. Kamath, I. Vulić, S. Ruder, K. Cho, and I. Gurevych. AdapterHub: A Framework for Adapting Transformers. 1, 2020. URL http://arxiv.org/abs/2007.07779.

[52] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych. AdapterFusion: Non-Destructive Task Composition for Transfer Learning. 2020. URL http://arxiv.org/abs/2005.00247.

[53] R. Steinberger, A. Eisele, S. Klocek, S. Pilos, and P. Schlüter. DGT-TM: A freely available translation memory in 22 languages. *Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC 2012*, pages 454–459, 2012.

[54] R. Rozis and R. Skadinš. Tilde MODEL - Multilingual Open Data for EU Languages. *Proceedings of the 21st Nordic Conference on Computational Linguistics, NoDaLiDa*, (131):263–265, 2017. ISSN 1650-3740. URL http://www.ep.liu.se/ecp/article.asp?issue=131{&}article=035.

[55] M. Artetxe and H. Schwenk. Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Transactions of the Association for Computational Linguistics*, 7: 597–610, 2019. ISSN 2307-387X. doi: 10.1162/tacl_a_00288.

[56] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, and B. Cowan. Moses: Open source toolkit for statistical machine translation. *Prague: Proceedings of 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions.*, (June):177–180, 2007. URL `http://dl.acm.org/citation.cfm?id=1557821`.

[57] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. pages 48–53, 2019. doi: 10.18653/v1/n19-4009.

[58] D. P. Kingma and J. L. Ba. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–15, 2015.

[59] G. Neubig and J. Hu. Rapid adaptation of neural machine translation to new languages. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 875–880, 2020. doi: 10.18653/v1/d18-1103.

Appendix A

A.1 Dataset Details

Language	Code	Train Set	Dev Set	Test Set
Romanian	ro	375976	4000	4630
Bulgarian	bg	383740	4000	4630
Slovenian	sl	594094	4000	4630
Hungarian	hu	594608	4000	4630
Polish	pl	600283	4000	4630
Latvian	lt	603327	4000	4630
Latvian	lv	608279	4000	4630
Slovak	sk	608864	4000	4630
Czech	cs	615801	4000	4630
Estonian	et	619805	4000	4630
Greek	el	1243699	4000	4630
Swedish	sv	1826250	4000	4630
Italian	it	1886848	4000	4630
German	de	1896440	4000	4630
Finnish	fi	1900165	4000	4630
Danish	da	1930805	4000	4630
Portuguese	pt	1937771	4000	4630
Spanish	es	1941571	4000	4630
Dutch	nl	1965015	4000	4630
French	fr	1988104	4000	4630
Maltese	mt	1824202	4000	4000
Croatian	hr	685201	4000	4000
Irish	ga	155572	4000	4000

Table A.1: Training, Validation and Test Sets size on the 23 languages $\rightarrow$ English. The code corresponds to the ISO 639-1 code of each language.

A.2 Complete Results when translating to/from English

X-en	Bilingual	Many-to-One			Many-to-Many			
		Fully Shared	HR languages adapters	All languages adapters	Fully Shared	Source adapters	Target adapters	Source+Target adapters
ro-en	46.21	47.44	47.69	48.23	45.76	47.04	46.61	46.99
bg-en	43.62	45.09	45.49	45.96	43.34	44.57	44.22	44.59
sl-en	41.69	43.42	43.55	43.8	41.48	42.73	42.44	42.82
hu-en	39.28	39.79	39.88	40.39	37.49	38.95	38.3	38.91
pl-en	39.82	39.07	39.3	39.5	36.84	38.2	37.81	38.54
lt-en	36.43	37.13	37.21	37.44	35.17	36.23	35.86	36.44
lv-en	40.98	41.5	41.84	42.08	39.49	41.15	40.57	41.05
sk-en	41.61	42.97	43.26	43.32	40.96	42.14	41.95	42.21
cs-en	42.85	43.7	43.94	44.23	41.77	42.81	42.71	42.82
et-en	39.88	40.66	40.69	41.08	38.13	39.82	39.11	39.76
el-en	49.02	48.55	48.88	49.13	46.62	47.66	47.23	47.7
sv-en	45.29	44.6	45.02	45.03	43.01	44.14	43.54	44.05
it-en	39.34	38.97	39.38	39.47	37.51	37.99	38.16	38.23
de-en	42.21	41.68	41.77	42.21	40.01	41.02	40.64	41.03
fi-en	37.33	36.34	36.5	36.74	34.37	35.47	34.98	35.42
da-en	45.8	44.98	45.03	45.46	43.23	44.39	44.02	44.42
pt-en	47.72	46.79	47.28	47.49	45.23	46.25	46	46.21
es-en	48.52	47.64	48.22	48.41	45.83	46.94	46.56	47.09
nl-en	42.8	42.26	42.42	42.9	40.82	41.76	41.32	41.93
fr-en	43.94	43.13	43.55	43.73	41.63	42.74	42.2	42.53
mt-en	67.68	59.94	59.81	61.91	58.14	60.88	58.42	60
hr-en	61.75	72.53	48.29	50.61	43.15	48.77	44.68	46.58
ga-en	81.58	48.46	72.84	76.91	67.07	76.38	69	72.28

Table A.2: Results achieved when translating from any language to English.

en-X	Bilingual	One-to-Many			Many-to-Many			
		Fully Shared	HR languages adapters	All languages adapters	Fully Shared	Source adapters	Target adapters	Source+Target adapters
en-ro	39.25	33.23	33.60	38.40	38.42	40.21	40.97	41.16
en-bg	40.23	39.78	40.03	41.31	38.12	39.00	40.79	40.36
en-sl	38.21	36.76	36.75	38.07	34.58	35.71	37.61	37.19
en-hu	29.20	26.96	27.00	28.43	25.38	26.16	28.21	27.67
en-pl	29.57	28.13	28.43	29.66	26.62	27.50	29.27	28.95
en-lt	27.22	26.26	26.21	27.49	24.66	25.47	27.13	26.84
en-lv	35.04	34.17	34.49	35.45	32.69	33.39	35.06	34.79
en-sk	36.09	35.31	35.37	36.51	33.66	34.52	35.87	35.63
en-cs	33.97	33.22	33.46	34.32	31.67	32.51	33.67	33.51
en-et	27.89	26.41	26.32	27.67	24.74	25.26	27.35	27.23
en-el	41.58	39.78	39.42	40.77	38.20	38.87	40.81	40.07
en-sv	43.19	40.05	41.69	41.80	38.63	39.45	41.54	41.13
en-it	36.68	34.57	35.48	35.59	33.10	33.76	35.35	34.89
en-de	34.56	31.89	32.69	32.75	30.40	30.91	32.35	32.30
en-fi	28.52	25.68	27.05	27.13	23.96	24.70	26.59	26.11
en-da	46.12	41.95	43.38	43.77	40.73	41.26	43.39	43.11
en-pt	44.98	43.00	43.89	43.98	41.65	42.20	44.02	43.50
en-es	46.35	44.25	45.18	45.14	43.21	43.82	45.38	44.84
en-nl	38.98	35.73	36.90	37.11	34.60	35.18	36.86	36.70
en-fr	45.23	43.42	44.52	44.40	41.97	42.68	44.43	44.03
en-mt	53.11	47.90	47.93	49.68	46.10	47.26	50.14	49.38
en-hr	52.64	39.41	39.50	44.58	36.92	38.06	44.22	42.12
en-ga	69.82	60.08	59.99	63.58	56.99	58.49	64.02	61.44

Table A.3: Results achieved when translating from English to any language

A.3 Zero-Shot Translation Complete Results

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	w/adapters	Fully Shared	w/Target adapters	Source+Target adapters
bg	-	-	-	-	-	-
cs	26.33	26.09	26.78	7.94	18.72	19.10
da	34.32	32.74	33.87	4.14	25.78	24.89
de	26.45	25.39	25.98	4.44	20.67	19.35
el	31.05	30.63	31.36	10.48	28.14	25.96
es	34.92	34.42	35.12	7.16	27.03	27.08
et	21.23	20.76	21.46	9.11	17.99	17.45
fi	22.07	20.67	21.76	5.70	15.66	15.31
fr	35.90	35.49	36.19	9.44	29.32	28.65
hu	23.24	22.04	23.18	7.11	19.73	17.30
it	29.57	28.48	29.35	4.09	22.81	18.74
lt	21.98	21.57	22.61	5.43	18.99	16.42
lv	27.47	27.01	27.81	10.67	24.39	22.94
nl	30.76	29.22	30.30	3.98	22.55	23.33
pl	24.05	23.42	24.58	14.26	20.96	20.05
pt	33.82	33.33	34.05	6.08	23.57	24.37
ro	29.77	25.53	29.47	14.22	26.03	25.94
sk	28.34	27.87	28.65	10.92	23.03	23.14
sl	29.32	28.74	29.48	7.27	24.99	20.93
sv	31.60	30.58	31.95	6.92	22.77	24.30

Table A.4: Zero Shot translation results - Source language bg. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.78	30.83	31.68	12.17	28.62	27.95
cs	-	-	-	-	-	-
da	33.17	31.74	32.83	3.22	24.63	25.42
de	26.21	25.34	25.78	3.61	20.21	20.68
el	29.75	29.02	29.71	8.17	25.77	25.15
es	33.93	33.28	33.93	6.30	25.99	28.09
et	21.29	20.53	21.50	8.54	17.97	18.37
fi	22.10	20.47	21.56	4.71	16.07	16.19
fr	35.05	34.45	35.37	9.24	29.30	29.00
hu	23.00	22.16	23.24	8.03	20.22	18.53
it	28.79	27.70	28.31	4.50	23.28	21.29
lt	21.79	21.32	22.40	3.90	18.45	17.20
lv	26.96	26.65	27.56	9.04	24.33	23.66
nl	30.46	29.08	29.91	4.43	23.17	24.55
pl	24.02	23.56	24.43	14.00	21.30	21.47
pt	32.62	31.86	32.61	4.90	23.42	24.82
ro	28.68	24.35	28.15	12.74	25.05	25.71
sk	29.41	29.41	30.07	12.29	26.44	26.95
sl	29.49	28.88	29.70	7.58	25.88	24.00
sv	31.19	30.04	31.13	5.93	22.42	25.10

Table A.5: Zero Shot translation results - Source language cs. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	31.40	30.73	31.7	26.85	28.78	28.44
cs	26.07	25.18	25.95	20.3	22.67	22.63
da	-	-	-	-	-	-
de	27.04	25.31	25.96	18.56	24.31	23.47
el	30.51	29.04	29.93	24.87	27.34	26.56
es	34.72	33.49	33.89	25.66	30.75	30.25
et	21.75	20.36	21.33	17.57	19.42	19.27
fi	23.28	21.03	22.07	16.03	19.24	18.73
fr	36.14	34.71	35.52	28.10	31.68	31.39
hu	23.14	21.57	22.56	17.65	20.46	19.30
it	29.79	28.00	28.61	17.84	25.07	24.07
lt	21.57	20.90	21.54	16.08	19.28	18.76
lv	27.23	26.19	27.71	22.90	24.98	24.38
nl	32.11	29.53	30.59	22.95	27.72	27.51
pl	23.70	22.64	23.58	18.83	20.83	20.40
pt	33.53	32.13	32.65	20.71	28.08	28.10
ro	29.21	24.45	28.29	25.15	27.10	27.69
sk	27.68	26.89	27.77	22.26	24.77	24.11
sl	29.17	27.85	28.73	22.57	25.96	25.16
sv	34.77	32.52	33.63	26.03	30.95	30.91

Table A.6: Zero Shot translation results - Source language da. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	29.61	28.83	29.88	24.14	27.20	26.47
cs	25.11	24.43	24.99	17.09	21.63	21.92
da	33.29	30.97	32.01	15.94	28.27	28.39
de	-	-	-	-	-	-
el	29.17	27.84	28.58	21.15	25.57	25.13
es	33.63	32.51	32.96	22.94	29.25	29.08
et	20.54	19.44	20.25	15.94	18.24	18.18
fi	21.71	19.85	20.63	13.16	18.09	16.98
fr	35.06	33.72	34.51	23.30	30.56	26.68
hu	22.69	21.39	22.48	15.94	20.38	19.05
it	28.75	26.84	27.52	14.49	24.69	23.01
lt	20.95	20.25	21.01	13.59	18.61	18.04
lv	25.54	24.90	25.69	20.69	23.53	23.15
nl	31.25	29.01	29.84	16.73	27.38	26.86
pl	23.21	22.01	22.92	18.02	20.76	20.28
pt	32.73	31.09	31.72	17.49	27.34	26.77
ro	28.07	23.38	26.94	23.04	25.45	26.15
sk	27.36	26.39	26.95	21.01	24.16	23.86
sl	28.19	27.10	27.89	19.73	25.27	24.23
sv	31.27	29.27	30.53	20.06	26.86	26.41

Table A.7: Zero Shot translation results - Source language de. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	33.59	32.70	33.53	26.66	30.91	30.08
cs	27.57	26.80	27.56	17.40	22.79	22.30
da	36.07	33.64	34.56	11.71	28.75	28.76
de	28.50	26.34	26.99	12.47	22.86	22.20
el	-	-	-	-	-	-
es	37.17	35.89	36.36	18.77	32.09	31.35
et	22.28	21.20	22.07	15.56	19.19	18.65
fi	23.77	21.81	22.71	12.80	18.93	17.92
fr	38.13	36.94	37.53	21.38	32.75	31.59
hu	24.31	22.92	23.86	15.01	21.24	19.60
it	31.21	29.39	30.13	11.37	26.25	23.74
lt	22.51	21.98	22.74	12.81	19.77	18.33
lv	28.63	27.41	28.46	21.37	25.39	24.67
nl	32.47	30.46	31.22	12.95	26.75	26.42
pl	24.89	23.82	24.84	18.61	21.60	20.93
pt	36.05	34.56	35.32	14.86	29.17	28.61
ro	31.50	26.45	30.44	24.58	28.56	28.87
sk	29.32	28.52	29.18	20.50	25.47	24.67
sl	30.70	29.59	30.29	17.25	26.82	25.15
sv	33.98	31.73	32.94	19.18	28.02	27.85

Table A.8: Zero Shot translation results - Source language el. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	32.59	31.85	32.75	26.77	30.11	29.48
cs	27.23	26.32	27.00	18.58	22.69	22.67
da	35.79	33.16	34.37	12.39	28.28	28.59
de	27.53	25.76	26.48	13.30	22.91	22.71
el	32.67	31.05	32.06	22.16	28.95	28.67
es	-	-	-	-	-	-
et	22.06	20.73	21.46	16.38	19.23	19.20
fi	23.29	21.03	22.15	13.75	18.72	18.16
fr	38.18	36.88	37.74	22.62	34.03	33.33
hu	23.75	22.38	23.47	16.38	21.09	19.97
it	31.87	30.01	30.87	13.85	27.85	25.47
lt	22.26	21.42	22.20	13.06	19.62	18.73
lv	27.71	26.80	27.82	21.39	24.95	24.39
nl	32.40	30.10	31.11	14.79	27.01	27.14
pl	24.46	23.75	24.66	18.88	21.99	21.42
pt	36.98	35.26	36.06	16.76	31.46	31.91
ro	31.65	26.37	30.43	25.70	29.58	30.11
sk	28.89	28.20	28.94	22.07	25.24	25.17
sl	30.22	28.79	29.79	19.21	26.41	25.26
sv	33.76	31.49	32.73	18.91	27.38	27.23

Table A.9: Zero Shot translation results - Source language es. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	29.39	28.87	29.72	14.68	25.68	25.35
cs	24.57	24.36	24.87	8.68	17.17	18.59
da	32.15	30.15	31.06	5.29	24.27	25.02
de	25.28	24.10	24.47	5.57	19.27	19.71
el	28.02	27.24	27.73	11.72	23.37	22.98
es	32.51	31.72	32.30	9.29	25.58	26.01
et	-	-	-	-	-	-
fi	22.70	20.73	21.69	7.39	17.94	16.81
fr	33.94	33.12	33.68	14.43	27.72	26.93
hu	22.86	21.23	22.33	10.72	19.29	17.33
it	27.74	26.37	27.18	5.93	22.16	19.02
lt	21.30	20.73	21.45	7.06	18.14	16.95
lv	26.73	26.00	26.86	12.86	23.50	22.81
nl	29.46	27.91	28.66	6.13	22.90	23.09
pl	22.90	22.18	23.10	14.73	19.53	19.01
pt	31.06	30.16	30.74	8.65	23.20	23.23
ro	27.22	23.00	26.55	15.21	22.95	23.36
sk	26.75	26.05	26.86	12.54	22.06	21.95
sl	27.77	26.54	27.38	9.64	23.01	20.74
sv	29.91	28.51	29.53	9.37	23.08	24.23

Table A.10: Zero Shot translation results - Source language et. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	27.61	26.39	27.29	16.69	22.86	22.69
cs	22.99	21.98	22.33	11.27	17.02	17.60
da	30.61	27.72	28.89	5.79	23.34	23.51
de	23.55	21.99	22.32	8.59	18.29	18.43
el	26.53	24.77	25.47	13.85	21.90	21.39
es	30.57	28.84	29.47	10.60	24.07	24.05
et	19.78	18.25	18.98	13.21	16.55	16.37
fi	-	-	-	-	-	-
fr	31.91	30.20	30.83	16.41	24.94	24.86
hu	21.48	19.59	20.48	12.42	17.80	16.39
it	26.57	24.68	25.20	7.85	20.19	18.38
lt	19.88	18.85	19.44	7.89	16.62	15.64
lv	24.12	23.18	23.93	16.07	20.86	20.49
nl	27.89	25.48	26.40	8.97	21.35	21.68
pl	21.28	19.85	20.78	14.32	17.06	16.70
pt	29.40	27.69	28.25	11.47	21.64	21.86
ro	25.34	20.76	24.32	17.59	21.28	21.66
sk	24.48	23.22	23.76	15.08	19.64	19.48
sl	25.57	23.83	24.58	12.11	20.51	19.66
sv	28.83	26.46	27.76	11.02	21.64	22.74

Table A.11: Zero Shot translation results - Source language fi. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.87	30.00	30.96	26.26	28.50	28.10
cs	25.18	24.56	25.16	19.83	22.37	21.90
da	33.99	31.11	32.21	17.58	27.60	27.70
de	26.26	24.45	25.00	16.35	22.39	22.05
el	30.58	28.89	29.73	23.65	27.87	27.42
es	35.53	34.11	34.60	24.31	31.70	31.57
et	20.53	19.38	20.27	16.37	18.32	18.46
fi	22.04	19.93	20.85	14.04	17.31	16.89
fr	-	-	-	-	-	-
hu	22.64	21.16	22.29	17.34	20.27	19.38
it	30.89	29.00	29.69	18.64	27.18	25.84
lt	20.96	20.17	21.01	14.90	18.87	17.93
lv	26.29	25.22	26.05	21.49	23.77	23.60
nl	31.10	28.89	29.88	20.04	26.50	26.31
pl	23.25	22.14	23.22	18.87	20.96	20.59
pt	34.67	32.92	33.55	20.66	29.57	29.65
ro	29.53	24.65	28.61	26.36	28.26	28.88
sk	27.08	26.36	27.12	22.24	24.33	24.04
sl	28.63	27.02	27.79	20.35	25.26	24.44
sv	31.78	29.48	30.76	20.57	25.71	25.97

Table A.12: Zero Shot translation results - Source language fr. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	28.81	28.27	29.1	13.59	24.88	24.54
cs	24.01	23.52	24.17	7.68	16.05	17.69
da	31.12	29.24	30.31	4.23	23.01	22.60
de	24.64	23.41	24.00	5.00	18.87	19.12
el	27.87	26.84	27.49	10.66	23.29	22.49
es	31.94	31.29	31.86	8.33	25.23	25.50
et	20.12	19.17	19.88	9.69	16.64	16.51
fi	21.54	19.81	20.75	5.33	15.88	14.64
fr	33.42	32.54	33.32	12.20	27.44	26.54
hu	-	-	-	-	-	-
it	27.43	26.20	26.98	5.94	21.90	19.19
lt	20.47	19.72	20.46	4.49	16.56	15.37
lv	24.89	24.30	25.30	10.53	21.51	20.90
nl	29.22	27.42	28.17	6.59	22.64	21.90
pl	22.51	21.55	22.52	14.54	19.04	18.34
pt	30.86	29.87	30.59	8.00	23.05	22.87
ro	26.65	22.42	25.94	14.70	22.33	22.95
sk	26.02	25.29	26.11	9.81	20.39	20.72
sl	27.01	26.10	27.05	8.00	22.37	20.81
sv	29.24	27.79	28.87	7.38	21.68	22.17

Table A.13: Zero Shot translation results - Source language hu. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	29.07	28.15	29.25	23.25	26.47	25.98
cs	23.34	22.70	23.47	16.14	20.21	20.02
da	31.23	28.83	29.96	8.78	24.74	25.13
de	24.47	22.69	23.31	11.72	20.55	20.17
el	27.98	26.76	27.71	16.39	25.60	24.88
es	33.43	32.45	33.11	17.44	29.54	29.53
et	18.93	17.77	18.60	14.28	16.96	16.59
fi	20.34	18.47	19.38	11.30	15.97	15.74
fr	34.83	33.74	34.54	18.85	30.84	30.65
hu	21.53	19.90	20.68	14.92	19.05	18.01
it	-	-	-	-	-	-
lt	19.54	18.72	19.63	12.19	17.61	16.76
lv	24.19	23.25	24.24	19.40	21.98	21.86
nl	29.18	27.00	27.93	12.44	23.89	24.16
pl	22.21	21.37	22.33	17.41	19.90	19.53
pt	32.80	31.20	31.93	14.12	27.08	27.46
ro	27.36	22.83	26.36	22.22	25.69	26.16
sk	25.15	24.17	24.99	19.53	22.37	21.96
sl	26.05	25.02	25.93	16.18	22.95	22.22
sv	29.18	27.18	28.34	14.96	23.06	24.06

Table A.14: Zero Shot translation results - Source language it. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	27.42	26.99	27.86	13.51	24.49	23.77
cs	22.66	22.31	22.89	8.96	16.76	17.14
da	28.67	27.34	28.08	5.51	21.90	21.83
de	22.76	21.95	22.28	4.38	17.20	17.39
el	25.63	25.16	25.57	10.08	21.67	21.52
es	29.91	29.22	29.90	7.85	23.91	24.17
et	18.97	17.88	18.71	10.27	15.70	15.65
fi	19.86	18.54	19.36	6.65	15.62	14.45
fr	30.79	30.49	31.17	13.44	25.68	24.89
hu	20.59	19.64	20.51	9.15	17.26	15.76
it	25.57	24.71	25.38	5.76	20.61	18.35
lt	-	-	-	-	-	-
lv	24.03	23.72	24.22	15.18	21.82	21.06
nl	27.00	25.77	26.63	5.95	21.19	20.99
pl	21.02	20.54	21.28	14.46	17.77	17.69
pt	28.73	28.05	28.78	8.20	21.78	22.02
ro	25.07	20.99	24.85	12.50	21.17	21.50
sk	24.42	23.54	24.33	12.53	19.88	19.81
sl	24.97	24.23	24.89	10.53	20.97	19.33
sv	26.92	25.92	26.64	8.78	20.62	21.26

Table A.15: Zero Shot translation results - Source language lt. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.49	29.80	30.70	17.98	27.51	26.96
cs	25.32	24.90	25.54	10.83	18.68	20.22
da	32.35	30.55	31.57	6.53	25.18	25.73
de	25.00	24.10	24.61	5.90	19.70	20.04
el	28.73	27.70	28.67	13.22	24.05	24.11
es	32.62	31.92	32.58	14.13	26.84	27.52
et	21.28	20.42	21.20	13.72	18.72	18.54
fi	22.18	20.49	21.40	9.14	17.83	17.35
fr	33.97	33.26	33.99	16.21	28.46	27.67
hu	22.92	21.59	22.50	12.09	19.45	18.14
it	28.08	26.69	27.57	8.51	22.88	21.04
lt	21.90	21.36	22.09	8.06	19.03	18.01
lv	-	-	-	-	-	-
nl	29.81	28.06	28.91	8.78	23.59	24.29
pl	23.24	22.51	23.51	16.95	20.28	20.16
pt	31.48	30.75	31.22	10.72	24.22	24.79
ro	27.82	23.57	27.18	17.60	24.01	24.60
sk	27.39	26.88	28.11	14.72	22.81	22.92
sl	28.17	27.39	28.11	13.89	24.49	22.81
sv	30.48	29.03	30.02	11.22	23.99	25.11

Table A.16: Zero Shot translation results - Source language lv. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.23	29.49	30.45	26.04	27.98	27.50
cs	25.47	24.75	25.20	21.23	23.01	22.69
da	34.50	31.92	32.92	26.61	30.70	30.10
de	27.05	25.62	26.04	22.01	24.64	24.25
el	29.54	28.25	28.96	24.96	27.02	26.47
es	34.20	33.02	33.71	28.42	30.88	30.57
et	20.77	19.34	20.40	17.32	19.2	18.81
fi	22.41	19.93	21.06	15.67	17.86	17.20
fr	35.64	34.25	35.08	28.92	31.36	31.11
hu	22.94	21.52	22.67	18.57	21.02	20.14
it	29.14	27.48	28.10	21.55	25.06	23.80
lt	21.39	20.51	21.24	17.04	19.29	18.78
lv	26.29	25.36	26.12	22.52	24.39	24.01
nl	-	-	-	-	-	-
pl	23.45	22.28	23.34	19.15	21.27	20.96
pt	33.20	31.56	32.27	25.70	28.78	28.43
ro	28.35	23.54	27.40	25.02	26.66	26.9
sk	27.38	26.43	27.20	22.88	24.64	24.34
sl	28.61	27.12	28.07	23.24	25.67	25.15
sv	32.22	30.11	31.34	25.08	27.70	27.88

Table A.17: Zero Shot translation results - Source language nl. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	28.98	28.80	29.95	6.69	25.83	26.08
cs	24.49	24.23	24.98	3.89	13.66	18.20
da	30.89	29.20	30.37	1.92	20.94	22.06
de	24.75	23.61	24.26	2.20	17.13	17.47
el	27.63	26.98	27.58	3.96	23.01	21.38
es	32.23	31.85	32.44	3.60	21.94	24.41
et	19.78	18.93	19.78	4.13	15.44	16.02
fi	21.02	19.35	20.35	3.45	14.43	14.31
fr	33.57	33.03	33.83	5.35	26.15	27.19
hu	21.79	20.96	21.77	3.79	18.26	16.43
it	27.95	26.79	27.55	3.04	21.09	18.98
lt	20.69	20.02	21.25	2.14	16.71	14.81
lv	25.18	24.71	25.66	4.49	21.65	20.98
nl	29.00	27.73	28.60	2.80	20.92	21.96
pl	-	-	-	-	-	-
pt	31.32	30.58	31.36	3.49	20.11	22.96
ro	27.02	22.87	26.51	5.85	21.75	22.32
sk	26.50	26.12	26.93	5.74	19.66	21.52
sl	27.23	26.74	27.4	3.44	22.21	19.33
sv	29.30	28.04	29.02	3.50	18.13	22.30

Table A.18: Zero Shot translation results - Source language pl. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	32.90	32.04	32.93	27.78	30.32	30.14
cs	26.95	26.18	27.06	19.40	22.96	23.27
da	35.92	33.05	34.24	14.46	28.43	29.04
de	27.75	25.96	26.40	15.20	23.49	23.29
el	32.46	31.02	31.73	24.45	29.39	29.08
es	38.12	36.59	37.38	25.42	34.35	34.99
et	21.65	20.55	21.41	16.16	18.89	19.08
fi	23.13	20.84	21.84	14.02	18.08	18.00
fr	38.63	37.30	37.81	25.30	34.28	34.18
hu	23.93	22.32	23.51	17.34	21.27	20.29
it	32.31	30.46	31.15	15.10	27.84	26.67
lt	22.24	21.43	22.34	14.76	19.93	19.09
lv	27.70	26.72	27.61	22.75	25.27	25.07
nl	32.23	29.77	30.90	15.62	26.61	27.04
pl	24.92	23.95	24.86	20.04	22.54	22.01
pt	-	-	-	-	-	-
ro	31.58	26.42	30.46	26.46	29.55	30.36
sk	28.96	27.88	28.60	22.60	25.44	25.52
sl	30.07	28.89	29.76	20.58	26.36	25.77
sv	33.38	31.11	32.36	18.99	26.61	27.54

Table A.19: Zero Shot translation results - Source language pt. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	32.92	32.51	33.69	19.64	30.73	29.75
cs	26.63	26.76	27.45	9.48	18.97	19.37
da	35.10	33.41	34.54	3.34	24.37	21.83
de	27.33	26.09	26.7	4.69	20.63	19.69
el	32.06	31.61	32.37	9.99	28.94	27.29
es	36.97	36.45	37.02	7.77	28.91	30.62
et	21.79	21.09	21.94	10.31	18.90	18.02
fi	22.62	21.21	22.29	6.80	17.56	15.93
fr	37.68	37.32	38.06	12.33	33.04	31.47
hu	24.02	23.09	23.97	8.61	21.09	18.23
it	31.09	30.08	30.81	5.11	25.58	21.75
lt	22.24	22.13	22.87	6.08	19.04	16.21
lv	27.68	27.5	28.3	11.79	24.81	23.55
nl	31.80	30.38	31.34	5.33	24.54	24
pl	24.69	24.01	25.23	16.3	21.72	20.45
pt	35.79	35.01	35.8	6.88	25.76	26.55
ro	-	-	-	-	-	-
sk	28.85	28.66	29.45	13.71	23.87	23.45
sl	30.17	29.75	30.48	8.39	25.8	21.97
sv	32.87	31.81	32.98	8.13	23.58	24.51

Table A.20: Zero Shot translation results - Source language ro. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.9	30.9	31.75	11.85	28.53	27.13
cs	27.42	27.24	27.84	8.57	20.86	21.35
da	32.8	31.54	32.51	3.05	23.76	22.77
de	26.27	25.4	26.02	3.69	20.12	19.59
el	29.46	29.09	29.58	7.57	25.54	24.63
es	33.68	33.3	33.86	6.37	25.74	27.07
et	21.24	20.41	21.58	8.47	17.65	17.69
fi	22.34	20.81	21.52	4.61	16.02	15.33
fr	34.96	34.71	35.2	9.12	29.35	28.39
hu	23.15	22.14	23.14	7.76	20.34	17.78
it	28.58	27.75	28.46	4.35	23.29	20.03
lt	21.74	21.46	22.21	4.04	18.64	16.37
lv	27.02	26.76	27.59	8.58	24.63	23.27
nl	30.51	29.11	30.07	4.22	22.88	23.12
pl	24.13	23.61	24.42	14.15	21.38	20.89
pt	32.64	32.62	32.74	4.98	23.14	23.28
ro	28.31	24.27	27.98	11.8	24.46	24.88
sk	-	-	-	-	-	-
sl	29.66	29	29.95	7.5	26.35	21.34
sv	30.94	29.97	30.83	5.91	21.71	23.45

Table A.21: Zero Shot translation results - Source language sk. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.58	30.65	31.57	10.19	28.06	27.00
cs	26.44	26.21	26.94	6.33	16.87	19.59
da	33.02	31.62	32.74	2.53	22.85	21.53
de	26.12	25.26	25.89	3.18	19.39	19.49
el	29.06	29.12	29.77	6.94	25.74	24.64
es	33.88	33.54	34.24	5.39	24.51	26.59
et	21.01	20.42	21.36	7.85	17.72	17.86
fi	22.21	20.60	21.71	5.09	16.00	14.97
fr	34.75	34.66	35.29	8.34	28.60	27.72
hu	23.26	22.18	23.39	7.78	20.28	17.48
it	28.40	27.54	28.27	3.89	22.64	18.59
lt	21.83	21.30	22.14	3.94	18.17	16.43
lv	26.82	26.24	27.29	8.66	23.76	22.94
nl	30.35	29.17	29.95	4.03	22.43	22.68
pl	24.05	23.5	24.44	13.19	20.72	20.42
pt	32.51	32.12	32.94	4.32	21.87	22.50
ro	28.48	24.30	28.11	9.51	24.30	24.70
sk	28.50	28.01	28.91	10.85	23.57	24.21
sl	-	-	-	-	-	-
sv	31.01	29.94	30.94	5.16	20.55	22.75

Table A.22: Zero Shot translation results - Source language sl. Rows indicate the target language.

	Pivot-Based Zero-Shot			Direct Zero-Shot		
	Bilingual	Fully Shared	adapters	Fully Shared	Target adapters	Source+Target adapters
bg	30.99	30.19	31.09	26.22	28.35	27.90
cs	25.64	24.93	25.89	20.64	22.60	22.41
da	36.90	33.93	35.14	23.48	32.59	32.65
de	26.82	25.14	25.74	19.81	23.62	23.39
el	29.95	28.58	29.33	24.93	26.95	26.63
es	34.48	33.07	33.70	26.52	30.49	29.98
et	21.29	20.16	20.99	17.46	19.38	19.05
fi	23.28	21.00	22.14	16.40	19.07	18.64
fr	35.93	34.45	35.09	28.20	30.70	30.84
hu	23.10	21.59	22.66	18.00	20.51	19.69
it	29.37	27.63	28.50	18.83	24.93	23.84
lt	21.14	20.63	21.44	16.46	19.23	18.58
lv	26.89	25.83	26.97	22.91	24.64	24.27
nl	31.31	29.10	30.28	23.27	26.69	26.79
pl	23.77	22.56	23.62	18.99	21.01	20.71
pt	33.17	31.68	32.48	22.55	27.91	27.69
ro	28.97	24.22	28.13	25.15	26.97	27.69
sk	27.59	26.68	27.52	22.77	24.69	24.19
sl	28.84	27.52	28.46	22.64	25.85	24.96
sv	-	-	-	-	-	-

Table A.23: Zero Shot translation results - Source language sv. Rows indicate the target language.