UNIVERSITÀ
DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE

ICT International Doctoral School

Multilingual Neural Machine Translation for Low Resource Languages

Surafel Melaku Lakew

Advisor:

Marcello Federico Amazon AI / Fondazione Bruno Kessler

Thesis Committee:

Alexandra Birch University of Edinburgh

Ondřej Bojar Charles University

Jörg Tiedemann University of Helsinki

April, 2020

Trento, Italy

Dedicated to my family – for their love, support, and vision.

ይህ አንብሮ መታሰቢያነቱ ለቤተሰቦቼ ፍቅር፣ ድጋፍ እና ህልም ይሁንልኝ።

❄ ❄ ❄

Abstract

Machine Translation (MT) is the task of mapping a source language to a target language. The recent introduction of neural MT (NMT) has shown promising results for high-resource language, however, poorly performing for low-resource language (LRL) settings. Furthermore, the vast majority of the $7,000+$ *languages around the world do not have parallel data, creating a zero-resource language (ZRL) scenario. In this thesis, we present our approach to improving NMT for LRL and ZRL, leveraging a multilingual NMT modeling (M-NMT), an approach that allows building a single NMT to translate across multiple source and target languages. This thesis begins by i) analyzing the effectiveness of M-NMT for LRL and ZRL translation tasks, spanning two NMT modeling architectures (Recurrent and Transformer), ii) presents a self-learning approach for improving the zero-shot translation directions of ZRLs, iii) proposes a dynamic transfer-learning approach from a pre-trained (parent) model to a LRL (child) model by tailoring to the vocabulary entries of the latter, iv) extends M-NMT to translate from a source language to specific language varieties (e.g. dialects), and finally, v) proposes an approach that can control the verbosity of an NMT model output. Our experimental findings show the effectiveness of the proposed approaches in improving NMT of LRLs and ZRLs.*

Keywords

[Neural Machine Translation, Multilingual, Low Resource, Zero Resource, Zero Shot, Transfer Learning, Language Varieties, Translation Style]

Acknowledgments

Today I look back at the day I joined the MT group at FBK. The kick-off meeting, choosing my research topic, the ups and downs, and the results of our work. I am humbled and thankful that I took this career path, and for the best that is yet to come.

First of all, I would like to thank my advisor Marcello Federico. For the opportunity to be his Ph.D. student, for the guidance, and collaborations that lead us to fruitful results, thank you. I would also like to thank Matteo Negri and Marco Turchi, their encouragement, ideas, productive discussions, and collaborations sustained me throughout the program, thank you. For MT group researchers: Luisa, Mauro, Nicola, and Roldano, thank you for all the discussions and working together, for walking me through the subjects I have asked and for the valuable time spent together in weekly MT meetings.

I am grateful to the former and current Ph.D. students of MT group with whom I had the chance to journey together: Amin, Mattia, Duygu, Rajen, Amir, Alina, and Marco, thank you for making the research productive and enjoyable. Thank you to the interns with whom I have had the opportunity to work with. My gratitude goes to friends and colleagues at FBK and the University of Trento. Thank you to all FBK and University of Trento staff, researchers, and students, particularly for those committed to providing services that are the backbone of our daily work.

Finally, I would like to express my sincere gratitude and appreciation to my beloved family, for their love, guidance, and sacrifice. This is for my mom (Yeshihareg Bekele) and my dad (Melaku Lakew). A very special thank you to Tsegaye, Tigist, Meseret, Genet, Almaz, Eleni, Rahel, Yonathan, and Milen. Psalm 23.

Surafel Melaku Lakew | April 2020, Trento, Italy.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Machine Translation (MT) is the task of automatically translating from a source language to a target language, by preserving semantic equivalence. MT has come a long way since the early approaches formulated by handcrafted rules to map lexical and syntactic structure across two languages [73], and the first data-driven rules of the example-based MT approach [73]. More recently, we have witnessed the transition from statistical MT (SMT) [86], in which translations are conventionally inferred from features (translation rules and beyond) extracted from the data to neural MT (NMT), in which the translation is directly learned from the data [27, 17, 119].

Both SMT and NMT models rely on parallel training data, where each data point is a parallel (source, target) sentence pair where the two sentences are the translation of each other. Recently proposed sequence-to-sequence (seq2seq) neural models have shown to work effectively for MT [79, 151, 31, 9]. They feature two main modules; an *encoder* that maps the source sentence into a hidden representation, and a *decoder* that utilizes the encoder representation to generate the target language sentence.

Recent advances in NMT have shown improvements over SMT both in fluency and adequacy [13, 14]. However, NMT performance is directly associated with the amount of available parallel data. When training data is reduced, the translation quality of NMT degrades faster than that of SMT [88, 14]. Moreover, large data requirements limit the applicability of NMT to less explored languages. A vast majority of the $7,000+$ languages [21] currently in use around the world have very limited data in the form of parallel examples and for this reason they are generally referred to as *low-resource*

languages. For some languages, the situation is even more complex since parallel data do not exist at all and MT has to operate in the so-called *zero-resource* condition.

Thus, based on the amount of available *parallel* data, we can identify two broad MT conditions; *high-resource* and *low-resource*. Moreover, for cases where there are no available parallel data for a pair of languages (but monolingual data), a *zero-resource* MT condition applies.

The broad interest towards MT in these complex settings is shown by the growing literature and variety of approaches to the problem. Most of these approaches are derived from notions such as *back-translation, transfer-learning, and multilingual modeling*, which represent the theoretical background of this thesis.

Back Translation, first used in SMT [15, 19], is a data-augmentation technique to generate synthetic parallel examples from monolingual data. Its effectiveness has been proved both in high and low-resource NMT settings [136]. Transfer Learning is an approach that spans across languages and models and is rooted in the linguistically motivated principle of *positive language transfer* [153]. In NMT, it can be applied to transfer lexical and grammatical information from a high-resource language to a related low-resource language [187]. Multilingual NMT (M-NMT) is an MT modeling approach, primarily aimed at building a single model to translate across multiple language directions [64, 77]. M-NMT can implicitly enable a positive language transfer across the languages represented in the single model. In addition to improving low-resource languages, an M-NMT model can enable "zero-shot" inference, that is a translation from source to target language which at training time have been only observed paired with another language [77].

This thesis, focuses on the limitations of NMT due to the amount of available training data when addressing NMT into low- or zero-resource languages, language varieties and styles. To tackle the lack of parallel training examples, our work is heavily based on M-NMT. In particular, modeling multiple translation tasks within a single model, leveraging better data augmentation techniques and maximizing positive transfer learning in order to incrementally improve *low-resource* and *zero-resource* MT represents the main theme of this thesis.

1.1 Research Hypotheses

This thesis investigates the following aspects regarding zero-resource languages, low-resource languages, language varieties and styles:

Low-Resource and Zero-Resource Neural Machine Translation with the use of multilingual models has shown positive results. In Chapter 3, we explore multilingual modeling in low-resource settings for various language directions. Then, we investigate the feasibility of zero-shot translation focusing on two dimensions: language relatedness and number of translation directions. From a modeling perspective, we empirically compare and verify the capabilities of the two prominent NMT architectures, recurrent and transformer, both in the low-resource and zero-shot translation settings. Chapter 3, ends by evaluating the hypothesis that multilingual modeling improves low-resource NMT and enables zero-shot translation. Chapter 4, evaluates the hypothesis that zero-shot translation can be further improved using a self-learning data augmentation technique that leverages monolingual data of the zero-resource pairs.

Transfer Learning for Low-Resource Languages has been highly beneficial to improve NMT performance. The two main variants of transfer-learning are: pre-training a high-resource model to initialize the low-resource model parameters, and training a single model using many available language pairs (M-NMT). We consider the following hypothesis: *i*) tailoring the transferred parameters to the low-resource languages, specifically focusing on their vocabularies, can result in a better positive transfer-learning; *ii*) instead of leveraging all the available data from different languages, NMT for a low-resource language pair can be improved by using the most relevant examples from the closely related languages. The evaluation results of our transfer-learning hypothesis are presented in Chapter 5.

Translation into Language Varieties and Styles is an important application of NMT, sharing multiple commonalities with low-resource MT modeling. However, the standard practice is to build separate translation models for different dialects and styles of the same language. As a result, training examples are split into several models, leaving certain dialects and styles in the low-resource setting. Our hypotheses builds on the principle of multilingual NMT. We aggregate data from different varieties or styles, and aim at preserving the positive transfer learning within a language. Particularly, we hypothesize

that our method will benefit the low-resourced variety or style and avoid the cost of building and maintaining separate models. In Chapter 6, we assess our hypothesis of building a single NMT model, for translating into different varieties (e.g. *dialects*) of the same language. Finally in Chapter 7, we evaluate the hypothesis of building an NMT model able to produce translations for a desired level of verbosity.

1.2 Contributions

The contributions of this thesis enhance the state of the art of NMT for low-resource and zero-resource language pairs. As a central unifying criterion, our research hypotheses build on top of a multilingual modeling, by leveraging the positive transfer learning across languages, language varieties, and styles.

Hence, the specific contributions in this thesis with referenced publications are:

- An in-depth evaluation of multilingual NMT approaches and comparison against single language pair models in low-resource and zero-shot translation conditions. Moreover, empirical comparison of the two prominent NMT architectures in single pair, multilingual, and zero-shot settings.

  - **Lakew, Surafel Melaku**, Mattia Antonino Di Gangi, and Marcello Federico. "Multilingual Neural Machine Translation for Low Resource Languages". In *Proceedings of the 4th Italian Conference on Computational Linguistics (CLiC-IT)*, Rome, Italy, 2017
  - **Lakew, Surafel Melaku**, Mauro Cettolo, and Marcello Federico. "A Comparison of Transformer and Recurrent Neural Networks on Multilingual Neural Machine Translation. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, New Mexico, USA, 2018

- A novel self-learning approach to improve zero-shot NMT model by utilizing only a monolingual data from a pair of languages to learn their dual (*source* $\leftrightarrow$ *target*) translation directions.

  - **Lakew, Surafel Melaku**, Quintino F Lotito, Negri Matteo, Turchi Marco, and Federico Marcello. "Improving Zero-Shot Translation of Low-Resource Languages". In *14th International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan, 2017
  - **Lakew, Surafel Melaku**, Marcello Federico, Matteo Negri, and Marco Turchi. "Multilingual Neural Machine Translation for Low Resource Languages". In *Italian Journal of Computational Linguistics (IJCoL)*, Rome, Italy, 2018

- A dynamic transfer-learning approach from a pre-trained model to a new language directions, that shows to significantly improve performance in low-resource conditions, and an extension with relevant data selection strategies for an efficient transfer learning.

  - **Lakew, Surafel Melaku**, Aliia Erofeeva, Matteo Negri, Marcello Federico, and Marco Turchi. "Transfer Learning in Multilingual Neural Machine Translation with Dynamic Vocabulary". In *15th International Workshop on Spoken Language Translation (IWSLT)*, Bruges, Belgium, 2018
  - **Lakew, Surafel Melaku**, Alina Karakanta, Marcello Federico, Matteo Negri, and Marco Turchi. "Adapting Multilingual Neural Machine Translation to Unseen Languages". In *16th International Workshop on Spoken Language Translation (IWSLT)*, Hong Kong, 2019

- A multilingual NMT approach to handle machine translation into different varieties of a language.

  - **Lakew, Surafel Melaku**, Aliia Erofeeva, and Marcello Federico. "Neural Machine Translation into Language Varieties". In *Proceedings of the Third Conference on Machine Translation: Research Papers (WMT)*, Brussels, Belgium, 2018

- An NMT modeling approach for machine translation into different language styles, in particular targeting the verbosity of the output.

  - **Lakew, Surafel Melaku**, Mattia Di Gangi, and Marcello Federico. "Controlling the Output Length of Neural Machine Translation". In *16th International Workshop on Spoken Language Translation (IWSLT)*, Hong Kong, 2019

The contributions and resources derived from the works in this thesis can be accessed in the following repository https://github.com/surafelml/phd-thesis.

1.3 Structure of the Thesis

This thesis is organized in eight chapters:

Chapter 1 introduces the motivation, problem areas, hypotheses, and lists the specific contributions discussed in this thesis.

Chapter 2 discusses MT in general, introduces neural networks for NLP, sequence-to-sequence models, recurrent and transformer models. It then focuses on NMT modeling using neural networks. Finally, it introduces multilingual NMT in connection with the approaches proposed in the following chapters.

Chapter 3 presents recent developments in multilingual NMT. Provides the pros and cons of multilingual modeling for low-resource and zero-resource languages. It is followed by a comparison of RNN and Transformer architectures for multilingual models.

Chapter 4 introduces and discusses a self-learning approach to improve zero-shot translation. Experimental results are analyzed in a multilingual setting.

Chapter 5 discusses a dynamic transfer learning approach that tailors a pre-trained model to a new language of interest. The approach is compared with a conventional adaptation method on the initial model parameters and configurations. Moreover, the proposed approach is evaluated in combination with a data selection strategy.

Chapter 6 presents an approach that extends the multilingual NMT principle to language varieties for which labeled and unlabeled data are available. Detailed analyses and comparisons are provided for multiple models trained on specific dialects and closely related languages.

Chapter 7 presents approaches for controlling the verbosity (length) of NMT output. Experimental settings are provided both for a low-resource and a high-resource setting, followed by a human evaluation controlling the quality level of NMT output, in the case of short translations.

Chapter 8 concludes by summarizing the contributions of the thesis, outlining the remaining open challenges and potential direction for future work.

Chapter 2

Background: Sequence-to-Sequence Model for Multilingual Translation

In this chapter, we begin with an overview of machine translation (MT), followed by a discussion of artificial neural networks (NNs) and the basic building blocks used for implementing MT models. First, we introduce the simplest form of NNs, called feed-forward NN. Then, while discussing the necessary ingredients and algorithm for training a NN, we introduce a more powerful NN variant, called *recurrent* NN. Hence, we discuss sequence-to-sequence (seq2seq) modeling − a NN connecting two distinct NNs referred to as *encoder* and *decoder*. We further elaborate on seq2seq modeling focusing on how and why a new module called *attention* is integrated with the *encoder-decoder* networks. This will lead us to discuss on a more recent variant called Transformer, a NN solely built on the notion of *attention* mechanism.

In the rest of the chapter, we focus on neural MT (NMT), by describing the required procedures to build a translation model, the training and inference stages, data preparation and model evaluation. We then look at the different NMT training variants, supervised, semi-supervised and unsupervised training. We conclude by relating motivations and hypotheses, discussed in Chapter 1, with low-resource and zero-resource NMT.

2.1 Machine Translation

The first known use of MT dates back to the 1930's in the form of a mechanical dictionary device between multiple languages [74].

Later, in the 1950's International Business Machines introduced a computerized translation for multiple phrases between English and other languages under the project name Georgetown–IBM [72]. From the 1950s to the 1980s, several approaches have been proposed, collectively called rule-based MT. The first is a *direct*-MT using a dictionary and a list of rules for pairing words from one language to the other, then a *transfer-based*-MT with translation rules at the level of grammar was introduced [73]. However, the use of these approaches was very limited. Following an *interlingua*-MT is proposed, that is based on mapping each language to an intermediate representation or meta-language (also known as *interlingua*), and building a linguistics rule between the interlingua and each of the languages. In this way, *interlingua*-MT was conceived to facilitate a cross-lingual translations, in other words it aims to create a multi-language translation system.

Note that the classic rule-based interlingua-MT is far from achieving a real interlingua, the illustration given here serves as a way of communicating MT progress. Though there is a progress from the direct-MT towards an interlingua-MT, an interesting observation could be the emergence of MT systems that instead of a single language direction translate between multiple languages.

The next two significant MT developments are the *example-based* MT and *statistical*-MT (SMT), also known as *corpus-based* methods for their reliance on a parallel corpus [75]. In example-based MT, the underlying assumption is translation by analogy, i.e. by applying translation patterns at the phrasal level, extracted from the parallel data, that matches the input [73]. With SMT, three major innovations are introduced: word alignment models to extract translation rules from parallel data, the use of probabilistic models, and the application of efficient search algorithms [73]. The most effective approach in SMT is called phrase-based SMT [86], which is the approach that showed to work most effectively until the introduction of neural machine translation (NMT) in the mid 2010s.

2.2 Artificial Neural Networks

Artificial neural networks (NNs) are inspired by simplified computational models of biological neural networks. In its base form, an NN is a node (neuron) computing a binary function of its input (weighted) connections. The first mathematical formalization of NN was by McCulloch and Pitts [107]. An artificial neuron was modeled with a binary threshold: if the weighted sum of its input connections exceeds a threshold then the output is 1, otherwise 0. Hence, the behavior of the neuron depends both on the input and the (learnable) weights and threshold.

There have been several development stages of NNs with different formalizations, all aiming towards better ways of learning functions or predictors from data. NNs are in general organized into layers of nodes of the same type, where the first layer is fed with external input, each upper layer is fed with the output of the previous layer, and the top layer produces the final output of the NN. The smallest form of NN is the *perceptron*, representing a network topology with a single layer. Feed-forward NNs (FNNs) stack multiple layers of perceptrons, named, input, hidden and output layers [59].

The dynamics of human communication is built on sequences of words and sounds, forming pieces of information. Thus, language translation can be seen as a mapping between source language sequences and target language sequences. FNNs were conceived to process input and generate output of fixed length, thus they are not the best model to handle sequences. Hence, recurrent NNs (RNNs) were developed to handle input and outputs of variable lengths [47] for tasks such as language modeling [109], part of speech tagging [181], and machine translation [151]. RNNs process a sequence in a step-wise way, one token at each step. At each step, an internal "state" is generated which is a function of the input and the state of the previous step. However, given their strict sequential nature RNNs can be hardly parallelized. More recently, Transformer NNs were proposed to address the limitations of RNNs, resulting in better performance and higher parallelism [168].

In the rest of this section, we expand our discussion on FNNs and RNNs and their training algorithms, we point out some drawbacks of the initial RNN models and introduce recent improvements.

2.2.1 Feed-Forward Neural Networks

The simple form of FNN consists of one hidden layer and an output layer. Information flows from the input all the way to the output. Each layer is made of several neurons. Each neuron of a layer is linked with all neurons of the following layer, thus making the NN fully connected. All input of a neuron are multiplied by associated weights and summed up to an offset or bias term. Then, a non-linear activation function is applied to the result before passing it to the next layer.

By assuming that all neurons of the same layer apply the same activation function, we can conveniently express the involved computation with vectors (bold small letters) and matrices (bold capital letters). Let us formally define our simple FNN. The first layer computes a hidden representation $\mathbf{h}$ by applying an affine transformation on the input $\mathbf{x}$, followed by a non-linear activation function σ.[1]

$$\mathbf{h} = \sigma(\mathbf{x}\mathbf{W} + \mathbf{b}) \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^{d_i}$, $\mathbf{W} \in \mathbb{R}^{d_i \times d_h}$, $\mathbf{b} \in \mathbb{R}^{d_h}$, $\mathbf{h} \in \mathbb{R}^{d_h}$, with d_i and d_h being the dimension of the input and the hidden layers. Then the output layer, applies another affine transformation on the hidden representation followed by an output function g:[2]

$$\mathbf{o} = g(\mathbf{h}\mathbf{U} + \mathbf{c}) \tag{2.2}$$

where $\mathbf{U} \in \mathbb{R}^{d_h \times d_o}$, $\mathbf{c} \in \mathbb{R}^{d_h}$, $\mathbf{o} \in \mathbb{R}^{d_o}$, with d_o being the dimension of the output layer.

The collection of matrices $(\mathbf{W}, \mathbf{U})$ and bias vectors $(\mathbf{b}, \mathbf{c})$ are referred as the network parameters (θ). When the network is trained on a specific task, the training algorithm is responsible to set the initial value of the parameters and to optimize them to get the best model prediction.

There are several types of non-linear activation functions which are also used as output functions. Sigmoid, tanh, and ReLU are among the most commonly used [59]. The sigmoid maps a real value z into the range [0,1], formally:

$$\sigma(z) = \frac{1}{1 + exp(-z)} \tag{2.3}$$

[1] if $\mathbf{v} = [v_1, v_2, ..., v_d]$, $f(\mathbf{v})$ is a shorthand for $[f(v_1), f(v_2), ..., f(v_d)]$

[2] e.g. also a non-linear activation function.

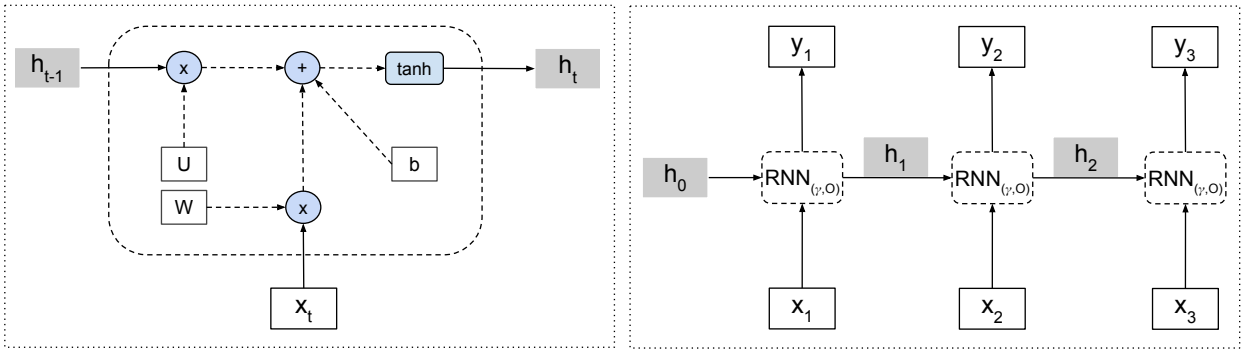

Figure 2.1: *Simplified version of RNN for a single time step (left), followed by the unrolled version (right) constructed by connecting each single RNN in a discrete-time step.*

Similarly, tanh (hyperbolic tangent) maps z into the range [-1,1]:

$$tanh(x) = \frac{exp(2z) - 1}{exp(2z) + 1} \tag{2.4}$$

A computationally more efficient alternative for training deep networks (NNs with many hidden layers) is the Rectifier activation function (ReLU). ReLU just clips the value z to 0 if $z < 0$;

$$ReLU(z) = \begin{cases} 0 & if z < 0 \\ z & otherwise \end{cases} \tag{2.5}$$

2.2.2 Recurrent Neural Networks

Differently from FNNs, RNNs can handle input and output of variable length. Let us consider a simple RNN for text tagging, i.e. with input and output sequences (words and tags) of the same length. As shown in Fig. 2.1 (left), for each input $\mathbf{x_t}$ an RNN computes a new hidden representation (state) $\mathbf{h_t}$ by linearly transforming the input and the previous hidden state $\mathbf{h_{t-1}}$, summing them, and finally applying an activation function:

$$\mathbf{h_t} = tanh(\mathbf{x_t}\mathbf{W} + \mathbf{h_{t-1}}\mathbf{U}) \tag{2.6}$$

where $\mathbf{x_t} \in \mathbb{R}^{d_i}$, $\mathbf{W} \in \mathbb{R}^{d_i \times d_h}$, $\mathbf{h_t} \in \mathbb{R}^{d_h}$, $\mathbf{U} \in \mathbb{R}^{d_h \times d_h}$, and d_i and d_h are the input and hidden

space dimension. The parametric formulation in Eq. 2.6 is called the Elman Network [47], or simple RNN.

The output vector $\mathbf{y_t}$, representing a probability distribution over the vocabulary of tags, can be computed by projecting the hidden state into the output space:

$$\mathbf{o_t} = \mathbf{h_t}\mathbf{V} \qquad \mathbf{V} \in \mathbb{R}^{d_h \times d_o} \tag{2.7}$$

and by finally applying the softmax output function to each component of $\mathbf{o_t}$:

$$[\mathbf{y_t}]_i = [softmax(\mathbf{o_t})]_i = \frac{exp([\mathbf{o_t}]_i)}{\sum_{j=1}^{d_o} exp([\mathbf{o_t}]_i)} \tag{2.8}$$

Notice that both our input and output at time t are vectors. We can think both of them as probability distribution over their respective word and tag vocabularies. Each input $\mathbf{x_t}$ is a so-called 1-hot vector, that has all the probability mass concentrated in the position associated to the specific vocabulary entry appearing at position t. While there is no uncertainty in the input distributions $\mathbf{x_t}$, each output $\mathbf{y_t}$ can in principle reflect the uncertainty of the NN about the correct tag. A simple criterion to generate a 1-hot output sequence is sampling from the $\mathbf{y_t}$ most probable tag at each step. This discussion about the nature of the output vector will be useful to understand the meaning of the loss function, that is used to compare network predictions of training examples against reference outputs, which are usually 1-hot vectors.

In the next section, we discuss how RNN can be trained from data.

2.2.3 Training Neural Networks

Training the simple RNN of the previous section requires a dataset C made of examples (X, Y) over which to optimize the RNN parameters θ, namely the three matrices $\{W, U, V\}$. The optimization criterion is to minimize an expected loss function over C, i.e:

$$\arg\min \; E_C[L(Y, Y(X, \hat{\theta}))] \tag{2.9}$$

where $Y(X, \hat{\theta})$ represents the output of the NN for input X and parameter values $\hat{\theta}$, and $L(Y, \hat{Y})$ is a convenient differentiable non-negative function such that $L = 0$ when

the prediction is correct ($Y = \hat{Y}$). Given the impossibility to find an analytic solution of Eq. 2.9 gradient descent methods are applied. Hence, starting from a random configuration of θ, the loss function expectation is computed, then its gradient is evaluated, and parameters are updated.

A crucial approach for simplifying the gradient computation is the computation graph of the NN. The computation graph represents the operations of a given network with a directed graph. In general, the computation graph abstracts and enables us to automatically build a network with input-hidden-output layers, evaluate the prediction results (i.e., loss), and update the parameters by computing the gradient with respect to the loss.

A loss function typically used for outputs which are probability distributions is the cross entropy. Given a prediction $\hat{Y} = (\widehat{\mathbf{y_1}}, ..., \widehat{\mathbf{y}}_\mathbf{L})$ and a reference $Y = (\mathbf{y_1}, ..., \mathbf{y_L})$ where each element of the two sequences is a d-dimensional probability distribution,

$$L(X, \hat{Y}) = -\sum_{t}^{L} \sum_{i}^{d} [\mathbf{y_t}]_i log[\hat{\mathbf{y}}_\mathbf{t}]_i \tag{2.10}$$

It is easy to show that if the reference is a 1-hot vector,

$$L(X, \hat{Y}) = -\sum_{t}^{L} log[\hat{y}_t]_{k(t)} \tag{2.11}$$

where $k(t)$ is the index of the correct label of position t.

For efficiency reasons, NN are trained with stochastic gradient descent (SGD) which approximates the expectation (Eq. 2.9) by taking a random sample of the training data, called mini-batch. The relevant details of the SGD-algorithm are shown in Table 2.1. The gradients of the loss function (Eq. 2.10) are computed on the computation graph of the RNN with the back-propagation algorithm [129, 97, 176]. The model parameters are updated towards the gradient and scaled with a learning rate (η) that determines the amount of the update.

Overall, a forward pass in the computation graph computes the output of each node in the topological order. This computation leads to the the final loss as in Eq. 2.10. Whereas the backward pass performs the back-propagation by computing the gradients in the reverse direction.

Given: data C, loss function L, initial parameter $\hat{\theta}$, learning rate η

While not *stopping criterion*:

$\hat{g} \leftarrow 0$

Sample mini-batch of size m, $[(X_1, Y_1), ..., (X_m, Y_m)]$, from C:

For i from 1 to m compute:

$L = L(Y_i, Y(X_i, \hat{\theta}))$

$\hat{g} \leftarrow \hat{g} + \frac{1}{m}\nabla_{\hat{\theta}} L$

Update: $\hat{\theta} \leftarrow \hat{\theta} - \eta\hat{g}$

return: $\hat{\theta}$

Table 2.1: *Algorithm 1: Mini-Batch Gradient Descent*

2.2.4 Long Short-Term Memory

RNNs are naturally deep in the time dimension, but can also be made deep by arranging multiple layer of simple layers. However, even simple-RNNs are prone to vanishing and exploding gradients.

Vanishing gradient is caused by loss gradients approaching zero [12]. In contrast *exploding gradients* occur when the computed gradients are very large. The exploding gradients problem can be addressed using a simple approach called *gradient clipping* [122]. Clipping works by limiting the norm of the gradient of the error ($\nabla_{\hat{\theta}}$) against a preset threshold. Simple-RNN are also not very effective in modeling long-term dependencies for a sequence of symbols, due to *vanishing gradients*, particularly for a long sequences.

The Long Short-Term Memory (LSTM) RNN was proposed to overcome this limitations. LSTM incorporates an additional memory cell for preserving gradients across time. Moreover, LSTM is better suited to capture long-range dependencies by enforcing the derivative of the recurrence function to be equal to one [70]. An LSTM cell has an input (i), output (o) and forget (f) gates, to compute how much to write and forget from the current memory-cell [58]. Hence, the RNN-LSTM computation for a state $\mathbf{h_t}$ is defined

using the current memory $\mathbf{c_t}$ and the previous hidden state $\mathbf{h_{t-1}}$, formally,

$$\begin{aligned}
\mathbf{g_t} &= \tanh(\mathbf{x_t}\mathbf{W_g} + \mathbf{h_{t-1}}\mathbf{U_g}) \\
\mathbf{i_t} &= \sigma(\mathbf{x_t}\mathbf{W_i} + \mathbf{h_{t-1}}\mathbf{U_i}) \\
\mathbf{f_t} &= \sigma(\mathbf{x_t}\mathbf{W_f} + \mathbf{h_{t-1}}\mathbf{U_f}) \\
\mathbf{o_t} &= \sigma(\mathbf{x_t}\mathbf{W_o} + \mathbf{h_{t-1}}\mathbf{U_o}) \\
\mathbf{c_t} &= \mathbf{c_{t-1}} \odot \mathbf{f_t} + \mathbf{g_t} \odot \mathbf{i_t} \\
\mathbf{h_t} &= \tanh(\mathbf{c_t}) \odot \mathbf{o_t}
\end{aligned} \tag{2.12}$$

$\mathbf{x_t} \in \mathbb{R}^{d_x}$, $\mathbf{g_t}, \mathbf{i_t}, \mathbf{f_t}, \mathbf{o_t}, \mathbf{c_t}, \mathbf{h_t} \in \mathbb{R}^{d_h}$, $\mathbf{W\_} \in \mathbb{R}^{d_x \times d_h}$, $\mathbf{U\_} \in \mathbb{R}^{d_h \times d_h}$. Here, $\mathbf{g_t}$ is similar to the simple-RNN operation in Eq. 2.6. The current memory $\mathbf{c_t}$ is computed using the forget gate $\mathbf{f_t}$ to determine the amount of $\mathbf{c_{t-1}}$ to preserve and the input gate $\mathbf{i_t}$ to decide the amount of new the update $\mathbf{g_t}$ to accept. Then, the current state $\mathbf{h_t}$ is computed by applying the tanh activation on $\mathbf{c_t}$, followed by a component-wise ($\odot$) product with the output gate $\mathbf{o_t}$. Overall, the computation of the hidden units using the gating operations allows the model to better represent long-range dependencies.

2.2.5 Gated Recurrent Unit

The Gated Recurrent Unit (GRU) simplifies the gating mechanism of LSTM. First introduced in [31], GRU avoids the need for a separate memory element and only uses a reset gate (r) and an update gate (z),

$$\begin{aligned}
\mathbf{r_t} &= \sigma(\mathbf{x_t}\mathbf{W_r} + \mathbf{h_{t-1}}\mathbf{U_r}) \\
\mathbf{z_t} &= \sigma(\mathbf{x_t}\mathbf{W_z} + \mathbf{h_{t-1}}\mathbf{U_r}) \\
\mathbf{\hbar_t} &= \tanh(\mathbf{x_t}\mathbf{W_h} + (\mathbf{h_{t-1}} \odot \mathbf{r_t})\mathbf{U_h}) \\
\mathbf{h_t} &= (1 - z) \odot \mathbf{h_{t-1}} + \mathbf{z_t} \odot \mathbf{\hbar_t}
\end{aligned} \tag{2.13}$$

$\mathbf{x_t} \in \mathbb{R}^{d_x}$, $\mathbf{r_t}, \mathbf{z_t}, \mathbf{\hbar_t}, \mathbf{h_t} \in \mathbb{R}^{d_h}$, $\mathbf{W\_} \in \mathbb{R}^{d_x \times d_h}$, $\mathbf{U\_} \in \mathbb{R}^{d_h \times d_h}$. In Eq. 2.13 we notice that the current state $\mathbf{h_t}$ is decided by an update gate $\mathbf{z_t}$. The update gate decides for each dimension whether to use the new candidate state $\mathbf{\hbar_t}$ or keep the previous one $\mathbf{h_{t-1}}$. The candidate is computed in a similar way as in simple-RNN, but with the additional term reset gate $\mathbf{r_t}$. GRU's formulation is much simpler in a way only the two gates are responsible for controlling the flow of information from previous states to the current

state computation. In overall, GRU-RNN assumes less parameters than an LSTM architecture. Moreover, empirical evaluations of LSTM and GRU show quite comparable performance [34].

2.2.6 Deep Networks

A deep network refers to a NN topology with multiple hidden layers stacked on top of each other. For instance, a *bi-directional RNN* constitutes two hidden layers dedicated to encode input sequences in a left-to-right and right-to-left fashion [9]. Based on Eq. 2.6, a two layer deep network can be formalized as,

$$\begin{aligned} \mathbf{h_t^1} &= \gamma(\mathbf{h_{t-1}^1}, \mathbf{x_t}) \\ \mathbf{h_t^2} &= \gamma(\mathbf{h_{t-1}^2}, \mathbf{h_t^1}) \end{aligned} \tag{2.14}$$

Notice that, the computation at the last layer of the current step $\mathbf{h_t^2}$ depends not only on the previous hidden state $\mathbf{h_{t-1}^2}$, but the current state of the lower hidden layer $\mathbf{h_t^1}$. The function γ can represent either the simple-RNN, the LSTM or the GRU recurrence. One of the main reasons for a multi-layer configuration is to model a network that learns both granular and more abstract features of the input sequence [112].

With the advantage of deep networks, however, comes the vanishing gradients problem in a layer-wise top-down back-propagation. Meaning, the bottom layers might not get a strong signal for learning. To address the drawback an approach called *residual connection* is suggested [68], which adds the output of the previous layer to the computation of the proceeding layer. By updating the previous Eq. 2.14, a state update at time t is formalized as,

$$\begin{aligned} \mathbf{h_t^1} &= \gamma(\mathbf{h_{t-1}^1}, \mathbf{x_t}) + \mathbf{x_t} \\ \mathbf{h_t^2} &= \gamma(\mathbf{h_{t-1}^2}, \mathbf{h_t^1}) + \mathbf{h}_t^1 \end{aligned} \tag{2.15}$$

Residual connections, effectively enable the network to extract features from the input in a progressive layer-wise manner and also propagate the information in a bottom-up direction.

2.3 Neural Machine Translation

Sequence-to-Sequence (seq2seq) is a neural architecture created by connecting two networks called *encoder* and *decoder*. Early seq2seq NNs have been proposed in several works [22, 54], however, they gained ground with machine translation (MT) [79, 151, 31]. Formally, MT can be defined as a task of mapping a source language sequence $X = x_1, x_2, \ldots, x_{L_x}$ with a target language $Y = y_1, y_2 \ldots, y_{L_Y}$, where the length L_x and L_y might be different.

Though there are different formalization of seq2seq models, the underlying working mechanism is similar. The encoder network reads the input sequence (X) and creates a latent representation of it, whereas the decoder learns how to generate the output sequence (Y). One way to model the encoder-decoder topology of a seq2seq is by using two separate RNN networks, hence, inheriting the ability to map arbitrary length source in the encoder and target sequences on the decoder.

In the following sections, we fist detail seq2seq model using only encoder-decoder networks, then we focus on an improved version that incorporates a new module known as *attention*.

2.3.1 Encoder-Decoder Model

The goal of an encoder-decoder seq2seq model is to map an input sequence X into the output sequence Y. Both sequences are feed into the network as a variable-length and transformed into a fixed dimension vector.

As shown in Fig. 2.2, first the encoder reads and transforms the source X into a real-valued fixed vector hidden state. Then, the decoder utilizes the encoded information to predict the target Y sequence. Extending the formulation in Eq. 2.6, an encoder-decoder network is formalized as,

$$\begin{aligned} \mathbf{h_t} &= \gamma^{(e)}(\mathbf{h_{t-1}}, E_S\mathbf{x_t}) && t = 1, ..., L_x \\ \mathbf{c} &= \mathbf{h_{L_x}} \\ \mathbf{z_t} &= \gamma^{(d)}(\mathbf{z_{t-1}}, E_T\widetilde{\mathbf{y}}_{\mathbf{t-1}}, \mathbf{c}) && t = 1, ..., L_y \end{aligned} \tag{2.16}$$

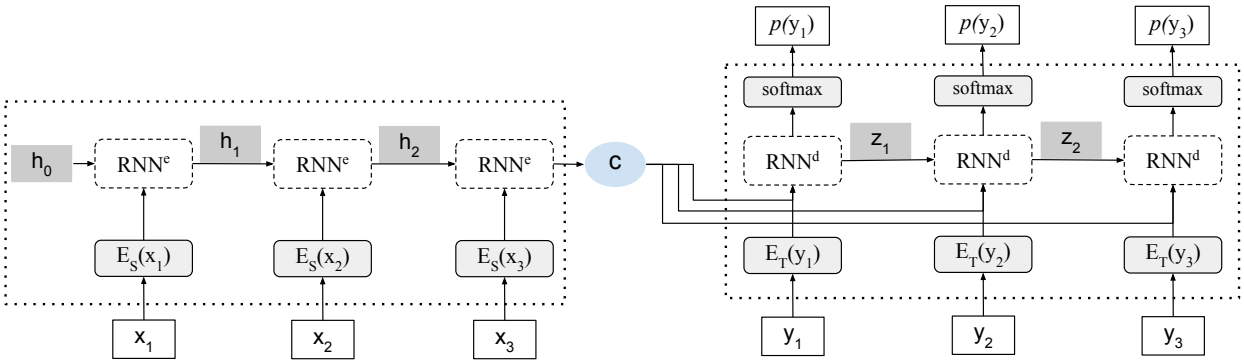

Figure 2.2: *RNN based encoder-decoder seq2seq model, with three input symbols (left), the context* $\mathbf{c}$*, and a decoder with three output symbols (right).*

In the first line of Eq. 2.16, the encoder RNN $\gamma^{(e)}$ computes the hidden representation of the source sequence X by looking up the encoder embedding $E_S\mathbf{x_t}$ at each time step t. Am embedding vector is the result of a projection of a sparse 1-hot vector into a smaller and more dense space. By the time the encoder generates the last hidden representation $\mathbf{h_{L_x}}$ (i.e., the context $\mathbf{c}$), all symbols in X are encoded.

Then, at each time step the decoder $\gamma^{(d)}$ computes the hidden state of the target symbols by looking up the embedding $E_T\widetilde{\mathbf{y}}_{\mathbf{t-1}}$. At time $t = 1$ the decoder is conditioned on the context of the encoder $\mathbf{c}$. In the subsequent steps ($t > 1$), the decoder takes the previous symbol representation $\mathbf{z_{t-1}}$ and the previously generated token ($E_T\widetilde{\mathbf{y}}_{\mathbf{t-1}}$) together with $\mathbf{c}$. The approach to utilize $\widetilde{\mathbf{y}}_{\mathbf{t-1}}$ for computing the state z_t is called an *auto-regressive* modeling [61].

Following, a softmax function takes the decoder hidden representation from each time step, to compute a probability distribution of the next target word (see right side of Fig. 2.2). The encoder-decoder model have shown impressive performance for complex sequence mapping tasks, such as MT [151, 31].

Bidirectional Encoder is built by stacking two hidden layers that read the input sequence in a forward and backward direction [9]. By combining the two representations, a bi-directional encoder learns a wider representation of the input. Thus, hidden representation at time step t is computed as;

$$\begin{aligned}\overrightarrow{\mathbf{h}_t} &= \overrightarrow{\gamma}^{(e)}(\overrightarrow{\mathbf{h}}_{t-1}, E_S\mathbf{x}_t) \\ \overleftarrow{\mathbf{h}_t} &= \overleftarrow{\gamma}^{(e)}(\overleftarrow{\mathbf{h}}_{t+1}, E_S\mathbf{x}_t)\end{aligned} \tag{2.17}$$

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}_t}; \overleftarrow{\mathbf{h}_t}] \tag{2.18}$$

At the last time step, the combined representation of the bidirectional encoder states forms the context ($\mathbf{c}$), which is used to condition the decoder network as in Eq. 2.16.

2.3.2 Encoder-Decoder with Attention

Attention is a module suggested to alleviate the drawback in the encoder-decoder model for efficiently representing long sequences [9]. The authors in [30] identified that a basic encoder can not efficiently compress long inputs in fixed size vectors. Hence, the *attentional encoder-decoder* network configuration is proposed starting from the bi-drectional RNN encoder. Then, a context vectors sequence set $(\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_{L_x}})$ is formed with the bi-directional encoder hidden states as in Eq. 2.18.

After the annotation of the source sequence, the decoder takes the combined encoder context vectors set $\mathbf{C}$ for generating the target sequence. Differently from the basic encoder-decoder approach (see Eq. 2.16)), the attention based decoder computation includes a time dependent context vector $\mathbf{c_t}$, in addition to the previous state $\mathbf{z_{t-1}}$ and previously generated token $\widetilde{\mathbf{y}}_{\mathbf{t-1}}$. The context $\mathbf{c_t}$ is the result of a computation from the encoder contexts $\mathbf{C}$ and an attention vector $\alpha_\mathbf{t}$ that tells the decoder how much emphases to give for each source word. Hence, at each time step t the decoder hidden state is computed as,

$$\mathbf{z_t} = \gamma^{(d)}(\mathbf{z_{t-1}}, E_T\widetilde{\mathbf{y}}_{\mathbf{t-1}}, \mathbf{c_t}) \tag{2.19}$$

Specifically the context vector $\mathbf{c_t}$, is computed as a weighted sum of the encoder states and attention weights:

$$\mathbf{c_t} = \sum_{i=1}^{L_x} \alpha_{\mathbf{t,i}}\mathbf{h_i} \tag{2.20}$$

The *attention weight* $\alpha_{t,i}$ is a normalization of the relevance score ($e_{t,i}$) over each context vector of the source sequence, formally,

$$e_{t,i} = f(\mathbf{h_i}, \mathbf{z_{t-1}})$$
$$\alpha_{\mathbf{t,i}} = \frac{exp(e_{t,i})}{\sum_{j=1}^{L_x} exp(e_{t,i})} \tag{2.21}$$

where i ranges over the source sequence $(1, \ldots, L_x)$, whereas the function f can be as simple as a scalar product [104]. The relevance score determines the importance level of the i^{th} encoder state for generating a target token at time step t.

2.3.3 Transformer Model

In RNN the hidden representations are generated in sequential order. In other words, computing the representation of input token x_t, for $t \geq 2$ is time dependent on the previous state computations $h_{<t}$. Here, the important question is how to address the challenges in RNN and leverage its strength in modeling long-range dependency. This is where the Transformer neural network (TNN) comes in, innovating on top of the attention mechanism introduced in [9].

2.3.3.1 Self Attention

TNN model avoids RNN's recurrence function relying on the principle of the attention mechanism. The newly proposed attention mechanism - known as *self-attention*, computes relations between the different positions of a given sequence to generate hidden representations. Hence, by removing the recurrence, self-attention enables parallel computation for the whole input sequence.

Similarly, as in RNN, a seq2seq TNN architecture is made up of an encoder and a decoder module, were both the encoder and decoder modules are entirely built using the self-attention mechanism. Partly analogous to the bidirectional RNN, both the encoder and decoder of TNN constitute a stack of self-attention layer followed by a fully-connected FNN layer. The encoder is composed of N number of similar layers. Each of the N layers

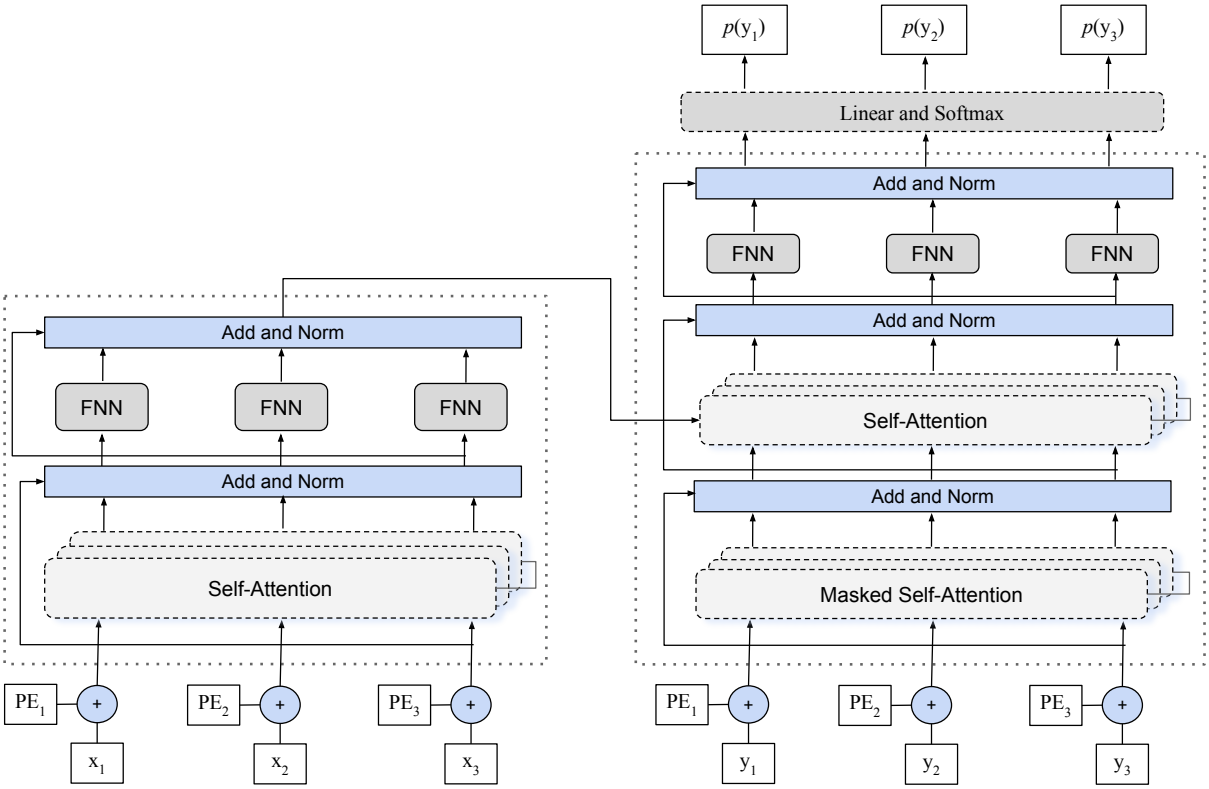

Figure 2.3: *Illustration of the Transformer model with a single encoder-decoder layer, three-headed self-attention and a position wise fully connected feed-forward sub-layers.*

comprises two sub-layers. The first sub-layer is a multi-headed self-attention, while the second is a FNN. The decoder side is similar to the encoder, except a third multi-head self-attention layer is added, to specifically attend on the encoder representation. Fig. 2.3, illustrates a single layer TNN in an encoder-decoder setup.

Recall in the attentional encoder-decoder RNN, the attention module allows the decoder to focus on the most important portion of the encoded information for generating tokens at each time step. Formally, the attention acts as a mapping function between a *query* from the decoder and *key-value* pairs in the encoder. The output score of the attention is the weighted sum of all the source input token values, where value is computed by a compatibility function of a query and the respective key. In TNN, the particular attention type is called a *scaled dot-product* attention, which is similar with a multiplicative attention type except it adds a scaling factor $\frac{1}{\sqrt{d_k}}$, where d_k is the dimension of the

key.[3] Assuming the scaled dot-product attention is computed on a set of queries (Q), and key-value pairs (K, V), that form three separate matrices, the operation is formally defined as:

$$\alpha(Q, K, V) = softmax(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}})\mathbf{V} \tag{2.22}$$

Multi-Head Self-Attention:
Unlike the formalization of attention in RNN, TNN is proposed to exploit multiple attentions simultaneously, also known as the *multi-head self-attention*. Each of the H numbers of attentions are differently learned dot-product attentions, simply concatenated before passing through a linear transformation. The authors in [168] showed that by linearly projecting Q, and K,V pairs H times, it is possible to jointly learn from the different representations of the sequences. In this modified attention definition, the queries and keys are of dimension d_k, while the value has a dimensionality of d_v; where $d_k = d_v = d_m/H$, d_m is the model dimension.

$$\begin{aligned} a_i &= \alpha_i(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V) \qquad i = 1, \ldots, H \\ A &= concat_a(a_1, a_2, \ldots, a_H)\mathbf{W}^O \end{aligned} \tag{2.23}$$

$\mathbf{W}_i^Q \in \mathbb{R}^{d_m d_k}$, $\mathbf{W}_i^K \in \mathbb{R}^{d_m d_k}$, $\mathbf{W}_i^V \in \mathbb{R}^{d_m d_v}$, are the projection matrices learned separately for H heads. Where as $\mathbf{W}^O \in \mathbb{R}^{H d_v d_m}$, is the linear projection of all the concatenated attention A from H number of attentions (a_i). An important attribute of the multi-head formulation is that computing H attention heads has a similar cost as computing single-head attention. This computational efficiency is achieved by minimizing the initial dimensionality of each head by the inverse of the headcount. Moreover, all of the attention functions α_i are computed in parallel, producing a d_v dimensional output.

In contrast to the RNN attention, TNN comprises three types of attention computations based on the origin of the query, and key-value pairs. The first is similar to the RNN *encoder-decoder attention* with the goal to attend on the important portion of

[3]Different alternatives for computing attention score are discussed in [104], in comparison with the additive attention proposed in [9].

the source sequence when generating each token on the target side. In this setting, the queries come from the decoder, and the key-value pairs are the output of the encoder. The second type of attention is computed in the *encoder self-attention* layers with the goal to build a representation of the input sequence. For the self-attention, the key-value pairs and the queries come from the previous layer in the encoder. In other words, each token in the encoder can attend to all the input positions in the previous self-attention layer. The third is the *decoder self-attention* computation which is similar to the encoder self-attention, except it only attends up to the current position.

Fully Connected FNN:
As shown in Figure 2.3, at the end of the multi-head concat operation, the information flows to a fully connected FNN sub-layer. The FNN operation is applied for each token and consists of linear transformations that are learned differently in each sub-layer, and a ReLU activation in the middle. Though the dimension of the FNN can be set differently (for instance; $d_{ff} = 2048$), the input and output dimension is kept similar with the model dimension, such as; $d_m = 512$. Note that, unlike the self-attention layers there is no dependency in FNN layers, hence, the computation for each symbol of the sequence can be done in parallel.

TNN maintains a similar output dimension for the embedding and each of the two sub-layers that are configured similarly with the model dimension. Moreover, before information flows to the next sub-layer the output is combined with its input using a residual connection [68], and layer normalization [8].

2.3.3.2 Positional Embeddings

Another important change in TNN is the introduction of position information to each of source and target token embeddings. Position information (also known as; *Positional Encoding (PE)*) is required due to the absence of time step dependent computation as in RNN, which resulted in the loss sequence order information. To compute the positional information, TNN utilizes a *sine* and *cosine* functions of different frequencies, formally:

$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{2i/d_m}})$$
$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{2i/d_m}}) \tag{2.24}$$

where *pos* corresponds to the current position, i is the dimension. Hence, every dimension constructs a sinusoid, with a wavelength progression. In this way, the PE is expected to learn even relative positions for cases where the decoder side requires offsetting. Before passing to the first self-attention layer, PE is easily summed with the embedding vector, since both are constructed with a similar dimension d_m. Note, there exist other variants of PE in the literature [57, 144], that showed close performance.

In this section, we focused on *seq2seq* modeling, including the basic RNN based encoder-decoder, followed by the attentional variant and concluded with the Transformer model. We examined how attention only approach can address some of the challenges in RNN for a more efficient sequence mapping task. In the next section, we focus on one of the complex seq2seq problem – machine translation.

2.3.4 Modeling

Several type of NN architectures have been proposed for modeling NMT: RNN [79, 151, 31, 9], Convolutional [57], and recently TNN [168]. In general, both RNNs and TNNs based NMT work with the same principle of an end-to-end training where gradients are back-propagated all the way to the encoder, however, there are subtle differences in sequence representation. RNN is the first seq2seq modeling approach that has been used successfully for MT task [151, 31, 9]. Recently, TNN [168] have shown better performance and efficient processing of input tokens in a simultaneous manner.

In this thesis, we utilized both RNN and TNN for experimental settings. We motivate our choice considering the increasingly effective performance improvements of NMT approaches in comparison with the previous standard, phrase-based SMT. Nevertheless, the improvements of NMT were limited to language pairs with a high amount of training examples. Hence, NMT for language pairs with small training data is a challenging and an open problem [88]. Most importantly, our choice of NMT is motivated by the promising

possibility of building a single model for multi-language translation [77].

2.3.5 Training and Validation

Given a collection of translation examples (*parallel data*) training of an NMT model is done by minimizing the expected cross-entropy loss on the data (Eq 2.9 - 2.11).

Model parameters are iteratively improve with SGD, by extracting random samples from the data, called batches. Actually batching often follows sampling based on sequence length for computational advantages. Different policies are used to adjust the learning rate of the SGD algorithm. The two primarily used methods are exponential decay [98] and Adam [82]. While the former applies a fixed and identical schedule on all the gradient dimensions, Adam and its variants apply adaptive scheduling on each dimension.

Regularization is a set of techniques that are mainly used to address model overfitting on training data, which mostly occurs in a low-resource training regime. *Dropout* is a common regularization approach that randomly sets to 0 some of the input connections to the neurons [150, 55]. Dropout is set with a pre-specified probability as part of the training configuration. It is also common practice to apply it differently on the output of each layer such as input, hidden and attention layers. [172] showed a relation between dropout and another regularization term L_2, a mechanism that adds the following penalty on the model weights $\hat{\theta}$, to the objective function to be minimized: $\frac{\lambda}{2}||\hat{\theta}||^2$, where λ is the scale set from the training configuration.

Stopping criterion is a requirement set to determine when a model training should stop. In literature, either a *manual* observation is made on the progression of the training performance or an *early stopping* criterion is set. Early stopping works by automatically checking the model performance over a given training interval against a minimum expected improvement threshold. Both the manual and automatic way of tracking the model performance is an integral part of NMT training, namely the *validation* stage.

2.3.6 Inference

At time of testing the NMT model reads sequences in the source language and generates the translations in the target language. The generation can be done by selecting the most

probable token at each time step, using a greedy search algorithm. A better solution is to search for an optimal translation by selecting at each step the top bw (beam width) candidates. Hence, its common to refer a bw of 1 as "greedy-search", and a $bw \geq 2$ as "beam-search".

When $bw = 1$, at each time step the most probable word is selected and the rest is discarded. While, with $bw = 5$, the scoring occurs by keeping the top 5 candidates, then the model produces a new candidate for each. All candidates are scored and the top 5 are kept while the rest is discarded. The operation is repeated until the model generates an end of sequence token (such as the pre-specified token $\langle eos \rangle$) [32]. In literature, a bw of $5 - 10$ is often applied.

Generally, the goal of the search algorithm is to find the target sequence Y that maximizes a score function $S(Y, X)$, formally: $S = logP(Y|X)$. However, S tends to favor short output over long once, since a $-log$ probability is added at each decoding step, assigning lower scores for longer sentences [178]. To address this challenge a length normalization term is utilized, which is now standard in NMT decoding,

$$\begin{aligned} S(Y, X) &= \frac{logP(Y|X)}{lp(Y)} + cp(X, Y) \\ lp(Y) &= \frac{(5 + |Y|)^{\alpha}}{(5 + 1)^{\alpha}} \end{aligned} \tag{2.25}$$

lp is the length normalization or length penalty term which is applied as an inverse factor to the scoring function. The term α is optimized on an evaluation set. Moreover, cp is a coverage penalty that utilizes the attention score on the source sequence (X), to favor translations fully covering the source sentence [178, 167].

2.3.7 Training Variants

In the previous section, we discussed NMT modeling assuming the availability of a parallel data. In this section, we discuss the different variants of NMT training.

2.3.7.1 Supervised

The assumption of a supervised NMT (s-NMT) modeling has been mentioned in the previous parts of this chapter. In simple terms, s-NMT requires the availability of a parallel training corpus. Hence, the objective function is simply to learn the mapping from the source and target training examples. Despite a change in modeling approaches (RNN or TNN), the training criteria for s-NMT stays the same, with the goal of minimizing the cross entropy loss function. Moreover, the (relative) size of training data defines if an s-NMT model is under the *low-resource* or *high-resource* category.

2.3.7.2 Semi-Supervised

In addition to the parallel data, monolingual data can be utilized for further improvements. The monolingual data can be available both for the source and the target languages. Hence, a semi-supervised NMT (ss-NMT) aims at utilizing monolingual data to gain further improvement over the s-NMT model. The primary way of achieving ss-NMT is known as *back-translation* [136]. To improve a direct source $\rightarrow$ target model with target language monolingual data, back-translation based ss-NMT training can be summarized in three steps:

i. Train a reverse target $\rightarrow$ source model using the available parallel data.

ii. Translate the target monolingual data with the reverse model.

iii. Train the direct NMT model by merging the original parallel data and the newly generated synthetic parallel data.

There are multiple ways of selecting the amount and type of monolingual data for back-translation. Two of the common approaches are random ratio based and similarity to the original data based selection [136, 175]. Recently, more variants of back-translation have been proposed. Adding noise with a certain probability in the synthetic source [46], alternatively, in [23] a tagged synthetic source as in multilingual NMT [77] have shown better improvements.

2.3.7.3 Unsupervised

Unsupervised NMT training assumes that parallel data is not available for the given language pair. Hence, unsupervised NMT (u-NMT) aims to learn just from monolingual data of the source languages. Though unsupervised training of SMT has been investigated before [128], works on NMT stated recently [94, 6]. The core idea of current u-NMT approaches share common principles, the first is learning a dictionary [36, 4], the second is to apply iterative back-translation on both directions $\leftrightarrow$ target directions. Hence, u-NMT [136] can be generalized in the following stages:

i. Learn a dictionary between the source and target, from the monolingual data.

ii. Generate dictionary based word-by-word translations for all monolingual data to form synthetic parallel data.

iii. Train an u-NMT model using the synthetic parallel data followed by multiple back-translation and training rounds.

The expectation is that through repeated back-translation and re-training the model will learn to produce better and better translations. In some u-NMT approaches, additional objective functions are with subtle modeling differences, such as: combining a generator (NMT model) and discriminator (language classifier) loss objectives [94]. Note that, an u-NMT converts the unsupervised task to a supervised (i.e., similar to ss-NMT training) version. Recently more variants of u-NMT have been suggested [182], even in combination with PB-SMT approach [5, 95]. However, there are findings that show underperformance of u-NMT when monolingual data are not comparable across source and target languages [63]. Moreover, u-NMT have shown not to work for languages that are characterized as *low-resource* and *distant* languages [113].

2.3.7.4 Multilingual

Multilingual NMT (m-NMT) maps sequences not only from a source to target language (single-pair NMT) but between two or more language directions. Since, the introduction of neural MT, several types of m-NMT approaches have been proposed [102, 43, 52, 99]. Here we focus on a single encoder-decoder based multilingual modeling approach proposed

in [77, 64]. To formalize the m-NMT model in [77], we simply extend the supervised NMT approach, with two key additions:

- The availability of a parallel training data from multiple (N) language directions.
- Pre-process each segment of the source side of each parallel data, by pre-pending a token (i.e., *language flag*) to identify the target language.

Then, the training data for m-NMT modeling is formed by simply aggregating the examples of the N directions. Hence, at training time the m-NMT model is optimized over the aggregated parallel data. Moreover, the loss function of the single-pair NMT model is utilized without any change. In the literature [77], learning shared model parameters $\hat{\theta}$, over the mix of N language directions have shown to be an effective strategy.

Multilingual Model Types:
Considering the number of language(s) used in the source and target side of the multilingual corpus, there are three types of multilingual modeling: *many-to-one, one-to-many, and many-to-many.*

The performance of each translation direction is subject to multiple factors (e.g. relative data size), as well as the chosen m-NMT model type. For instance, many-to-one modeling is easier than many-to-many, since the latter is also required to disambiguate the target language at inference time. As such, for L number of languages, the many-to-many setting can grow to a max of $N = L(L-1)$ directions. Regardless, how far the number of directions N can grow, an interesting question is how an m-NMT model can be utilized to address the challenging problems in MT, such as, low-resource language pair translation.

Universal Multilingual Model:
In principle, universal m-NMT can be considered as a branch or extension of the many-to-many setting. However, the term *universal* has been used for several NMT approaches that are applied in a language-independent modeling, or simply considering a large number of language directions in a single model. Indeed, in addition to better performance, the single encoder-decoder based multilingual model [77, 64] can accommodate more pairs with minimal additional model parameters and training complexity. Whereas, previously

suggested approaches have at least encoder and decoder modules that grow linearly with the number of translation directions [52].

Hence, we consider the universality of a model along at least three dimensions. First, the flexibility of the proposed approach and its applicability to any language without a language-specific modeling requirement.[4] Second, the type of m-NMT model in terms of directions (i.e., many-to-many), and the variants of languages included. Language variants can span different families, scripts and styles. Third, the capability of the multilingual architecture to model the representation of each language direction in a single semantic space.

There are quite a few works fulfilling all the above three criteria, and among them the single encoder-decoder m-NMT approach [77] provides the necessary platform for training a truly universal model. Hence, in this thesis, the *language-flag* based m-NMT modeling is used extensively. Note that, modeling a universal m-NMT is an open research problem that is still being investigated. Previously, [164] trained a single m-NMT model on Bible translations on over 900 languages, to learn a single universal representation space. Where as [1, 113, 160] trained massive models on over 55 language pairs for a low-resource translation task. More recently, the work in [3] have set a new milestone in universal m-NMT modeling, by designing and training a single model for 103 languages, with training examples in the magnitude of 10^9. Overall, since the premises in this thesis is built on top of m-NMT modeling, we will revisit the different approaches and the specifics through our discussion.

2.3.7.5 Zero-Shot

In spite of its success, zero-shot NMT (zs-NMT) has not been formally defined as a distinct modeling approach. Rather in most works, zs-NMT has been considered as part of m-NMT approach. The main reasons could be, the blurred boundary between zero-shot translation and m-NMT modeling, and the fact that the first successful zero-shot translation has been shown using an m-NMT model [77]. In this section, we first give clarity to the terms zero-resource translation, zero-shot translation, and to the respective modeling approaches. Then, we conceptualize zs-NMT and provide the specific reasons

[4]In this definition of *universality*, we avoided to examine the specific requirement of pre-processing tools, and only consider the availability of each language pair data.

behind positioning the condition as a distinct NMT modeling method along with the previously discussed variants.

Following the early works of neural-based m-NMT [52], a zero-resource modeling approach is proposed in [53]. The approach relies on a third *pivot* language to overcome the absence of parallel data for the direction of interest, hence, the name zero-resource NMT. Assume the languages i and j do not have a parallel training examples, and let both i, j have parallel data with the *pivot* language. By utilizing a pre-trained initial m-NMT model as in [53], a zero-resource NMT for the $i \rightarrow j$ translation direction is formulated as follows,

i. Translate into language i, the *pivot* side of the ($j - pivot$) parallel data.

ii. Using the translated data and the j side of the ($j - pivot$) parallel data, construct a pseudo parallel corpus for the zero-resource $i \rightarrow j$ direction.

iii. Fine-tune the m-NMT model with the $i \rightarrow j$ pseudo parallel data to build the zero-resource NMT.

Finally, the translation task for the direction $i \rightarrow j$ using the fine-tuned model is known as *zero-resource* modeling. Similarly, the first proposal of zero-shot translation shares the same principle of enabling a translation between the pair i and j, however, leveraging the *pivot* and/or other languages implicitly within the initial m-NMT model. Nevertheless, the first attempt of a direct zero-shot did not achieve a meaningful result in [53].

Formally, *zero-shot translation* refers to translating into a target language (j) from a source language (i), using an m-NMT. The m-NMT model is trained with data that does not include parallel training examples for the pair (i, j). The first effective zero-shot translation approach [77], leverages a multilingual model trained on a corpus that involve the language i and j paired with other language(s) (for instance, the *pivot* language as in zero-resource NMT). Moreover, the m-NMT modeling in [77], promising results for zero-shot translation is attributed to the fact that all the N languages share a single representation space, which is missing in [53].

Following, the success the direct zero-shot translation, newly proposed approaches open the door for zero-shot NMT modeling. In [161], a zero-shot NMT modeling is for-

mulated following a similar inference stage as in the zero-resource approach [53], however, with the key difference of translating directly between the zero-shot language pairs $i \leftrightarrow j$. Moreover, the approach incrementally improves both translation directions with progressive training and inference stages, which leads to a distinct modeling approach − *zero-shot* NMT. Hence, a working zero-shot modeling approach can redefine the universality of a multilingual model, by expanding the basic assumption of translating between language pairs included in the parallel data to language pairs without training examples. While further details of zero-shot NMT modeling are left for the next section, we summarize below the discussion on variants of NMT.

In relation to other variants of NMT, zero-shot modeling relies on a third language, the pivot language (e) or any other multilingual data, and an m-NMT model for the first inference stage to bootstrap the zero-shot translation model, while u-NMT relies on a dictionary induction and an initial word-by-word based translation model to bootstrap the unsupervised translation model. Both approaches rely on an iterative *back-translation* approach as in ss-NMT, and aim to improve over a loop of training and inference stages.

Overall, zero-shot, zero-resource, and unsupervised NMT modelings are based on the same assumption, where for a direction of interest there is no parallel training data available. Moreover, all these approaches aim at improving NMT models based on pseudo-parallel training examples.

2.4 Evaluation Metrics

To automatically evaluate the performance of MT output (*hypothesis*), metrics relying on gold standard (*reference*) translations are used. Here we give an overview of the automatic metrics that are used in our experimental settings, namely: Bilingual Evaluation Understudy (BLEU) [121], and Translation Error Rate (TER) [149].

BLEU measures the quality of a hypothesis based on its similarity with one or more references. The metric that defines BLEU is a modified *n-gram* precision multiplied by a brevity penalty that penalizes short outputs. The precision is computed as:

$$p_n = \frac{\sum_{U \in C} \sum_{ngram \in U} \hat{c}(ngram)}{\sum_{U \in C} \sum_{ngram \in U} c(ngram)} \tag{2.26}$$

where U is set of all unique n-grams in a sentence that belongs to the corpus of all the translations C, $\hat{c}$ is the $min(c(ngram), c_r(ngram))$, $c(ngram)$ is the count of ngram, whereas $c_r(ngram)$ being the count of $ngram$ in the reference sequence.

Then, the brevity penalty (BP) for a candidate translation length (l) and for a reference corpus length (r) is given as,

$$BP = \begin{cases} 1 & if\ l > r \\ e^{(1-r/l)} & if\ l \leq r \end{cases} \tag{2.27}$$

Finally, BLEU is computed as,

$$BLEU = BP * exp(\sum_{n=1}^{N} w_n log p_n) \tag{2.28}$$

where, N is the ngrams considered, w_n is the positive weight associated with p_n and summing to 1.

TER evaluates the minimum number of edits required to make an hypothesis equal to a reference. This number is then divided by the average length of the references. The edit types include four main operations, these are: shift (matches after displacements), insertion, deletion, and substitution.

Both BLEU and TER scores are reported after being normalized within the $[0 - 100]$ range. For BLEU the higher is better, whereas for TER lower is better. In Chapter 3, we show how we used TER, in a multi-reference setting (mTER) and in a lemma-based edit (lmTER) operations. Although we utilize BLEU in all of the experimental settings, in Chapter 7, we show how BLEU varies with respect to the BP for shorter translations.

2.5 Data Preparation and Processing

In this thesis, we will use *parallel, bi-text, bilingual* interchangeably to refer to translation examples for a pair of languages. In a multiple language setting, we refer to *multi-way*

parallel if the data is aligned across three or more languages. In cases where we have distinct parallel data for each pair of languages, we simply refer to them as *multilingual parallel* data. The other type of data we will use is *monolingual*, which is available per language.

Before training any NMT model, we split the available parallel data into three portions: *training, development, and test* sets. The training set is used to train the model parameters $\hat{\theta}$ by minimizing the loss L over training loops. The development set can be used at training time to set hyper-parameters (such as: learning rate, regularization, dropout), for the stopping criteria, and for selecting the best performing model. The test data is used to evaluate the final model performance. After the data split there can be several stages of data pre-processing, which we will summarize below.

Data cleaning can be done using several mechanisms. In standard NMT training the raw data passes through a set of common cleaning stages. The most common technique to clean C is to discard sentence pairs for which the relative difference in length is above a given threshold. For the above and other cleaning tasks, MT literature utilizes standard scripts from the Moses SMT [86], toolkit.[5]

Tokenization is a procedure that separates words from punctuation. Tokenization is an important part of the pre-processing stage. However, it is highly language-dependent, since words and punctuation are different from one language to the other.

Vocabulary is generated from the training data by taking the unique tokens along with the occurrence count. A vocabulary (V) can be generated separately for the source (V_S) and the target (V_T), or as a single combined list (V_{ST}). Based on the size of the data large vocabulary impacts the number of embeddings, softmax operations but also data sparseness and the softmax layer.

Word segmentation is a simple yet most effective approach to reduce the vocabulary size and address the out-of-vocabulary issue in NMT. The prominent approach is Byte-Pair-Encoding (BPE) [137] that splits rare or infrequent words into sub-word units using a model learned on the training data based on the frequency of character n-grams. More recently, SentencePiece (SP) an alternative approach to BPE is suggested [91]. Unlike BPE, SP emphasized to train the segmentation model on non-tokenized raw data. SP also

[5]Moses SMT: https://github.com/moses-smt/mosesdecoder

has the option of training using BPE algorithm or a unigram language model. Both in BPE and SP, the vocabulary size is pre-determined. Following the training of the model, the segmentation is applied to the three portions of the corpus accordingly.

2.6 Multilingual Neural Machine Translation

In the previous sections, we discussed the working principle behind neural networks, focusing on human language processing. Specifically, we identified MT as a complex mapping problem between a source and target language sequences. We reviewed the progress of MT from rule-based approach to the current standard, seq2seq architectures. In seq2seq modeling, we discussed the two prominent approaches for NMT: RNN (recurrence) and TNN (self-attention). Then, we covered the different types of NMT modeling, training, validation, testing, data processing, and evaluation metrics.

In this section, we expand our discussion on NMT variants, particularly multilingual NMT. We begin by highlighting the current understanding and motivation behind multilingualism in MT research. Then, we make a connection between multilingual modeling and the hypotheses made in the chapters to follow.

Before discussing each hypothesis, we first define the notion of language, language varieties, and styles. Then, we focus on the challenges of building an NMT model for languages with very limited parallel data, and for languages with only monolingual data. Finally, we focus on the problem of addressing subtle translation requirements, such as translating into a specific dialect of a language and a pre-determined output style.

Though the amount is different, NMT data is provided in two main forms: parallel and monolingual. Recall, we begin this chapter by describing the early notion of modeling human languages using neural networks. Throughout this thesis, in addition to language, we use the term language varieties, and styles. Hence, we found it important to formally define these terms, before discussing the variants of NMT training using the two forms of data,

- Language: "is what the members of a particular society speak" [174], or more generally its a method of communication in written or spoken forms.

- Language Variety: is a form of language, that may include dialects, registers, styles, and other forms of language, as well as a standard language (e.g. *Brazilian Vs. European Portuguese*.

Note, the degree of similarity between the varieties of a language vary depending on several factors. One way to determine the relatedness between languages is to identify the linguistic characteristics, such as vocabulary, grammatical structure, writing script. See [174] for a more comprehensive discussion.

Back to an Interlingua Modeling:
Firat [51] discussed the notion of multilingual translation using a single medium, and by finding a principled base in the Shannon and Weaver theory of communication [143]. Shannon and Weaver define communication to involve three parts; sender, receiver, and channel [143]. Accordingly, [51] draws a parallel between a Shannon and Weaver communication problem and seq2seq mapping that constitutes an encoder, decoder, and interface modules. Moreover, [51] explains the principles of interlingua modeling in relation to Weaver's Memorandum for machine translation [20] − with the analogy of the "tall towers". The Weaver analogy imagines communication between individuals living across a street in tall closed towers, in two forms; *i.*) either communication is done by shouting back and forth from each tower, leading to a poor information flow, *ii.*) or by descending to the common basement to establish an efficient and better communication. With the analogy, Weaver argues, a translation should not be restricted to a $source \rightarrow target$ direction, perhaps, discovering the common base of human language communication is a way forward. Likewise, Firat [51] regard the common base of communication as the interlingua, which is formulated as a *shared medium* with one of the first seq2seq architecture based multilingual model [52].

As such, [51] emphasizes on the question of modeling the shared medium. Where in case of an increasing number of language, the number of encoders and decoders increases while keeping a single shared medium [52]. Recall, recent advances in multilingual modeling avoids the need for such multiple encoder and decoder per language direction. We also mentioned, the simple idea in [64, 77] for a multilingual model is the additional *language-flag* on the source side. Hence, based on the multilingual modeling assumption in [52] with respect to Weaver's analogy, the shared (single) encoder-decoder architecture in [77] is a step closer to a true interlingua modeling.

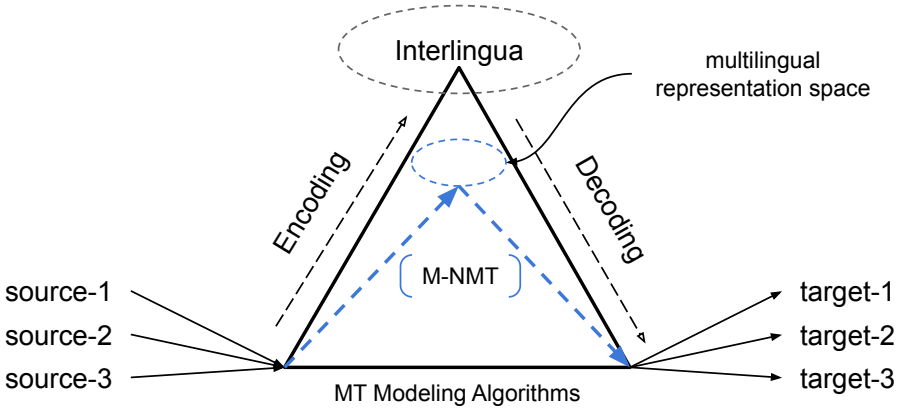

Figure 2.4: *Re-imagining Vauquois Triangle in a Multilingual NMT setting, where an interlingua representation space is formed from multiple languages.*

Furthermore, the idea of modeling multilingual MT was among the early approaches, long before the introduction of seq2seq architecture. Recall, our discussion on interlingua-MT in Sec. 2.1, where multilingual translation is first conceptualized by mapping each language to an intermediate representation (i.e., meta language) using rules [73]. If we consider seq2seq modeling (particularly for m-NMT), we can find a connection with the early principle of an interlingua-MT – *learning a common meaning or semantics representation space for a cross-lingual MT task*. In multilingual NMT, the representation space is learned by encoding different language sequences in a common latent space. Likewise, we adapt the *Vauquois triangle* [169] (see Fig. 2.4), to visualize m-NMT abstraction of the common semantic space.

In this conceptualization of m-NMT, the seq2seq modeling is the common learning algorithm for all the languages in the source and target side. Unlike what is proposed in the early works of interlingua-MT, NMT allows computing a latent representation to effectively map the source and target sequences, without language specific rules. The top section of the triangle in Fig. 2.4 depicts the interlingua representation, relating to the early conceptions of semantic level representation for multiple languages, however, using a seq2seq model. Similarly illustration of m-NMT for finding the emergence of continuous language space can be found in [164]. Note that, we do not claim that the multilingual semantic space is similar to the interlingua at the top of the triangle, instead we emphasize the principle of m-NMT model progressively leading to an interlingua.

Indeed, a single representation space in a multilingual model have shown to be advantageous, by enabling multiple translation directions. Moreover, the most interesting aspect comes from the possibility of leveraging a single semantic space for different type of translation problems. In the following sections of this chapter, we revisit the hypotheses given in Chapter 1 and capitalize on a single encoder-decoder based multilingual translation modeling. We provide a summary of the challenges in NMT with respect to our hypotheses and discuss the context of the proposed approaches in this thesis.

In particular, we emphasize on the following main questions which we base our premises for further exploration.

i. How to improve translation of a low-resource language pair with multilingual data?

ii. How to enable translation between a source and a target language without explicitly having or training on parallel data?

iii. How to incrementally improve a language direction without parallel training examples in a multilingual model?

iv. Can we achieve a better transfer-learning from a pre-trained multilingual model to a low-resource language pair?

v. Can we use the same principle of multilingual modeling to address the translation problem into different language varieties and styles?

In the following sections, we provide a summary of the challenges in NMT with respect to the above five questions, and the context of the proposed approaches in this thesis.

2.6.1 Low-Resource and Zero-Resource NMT

In the past few years, NMT has shown remarkable improvements, outperforming long established PB-SMT [86] approaches [16, 18], this improvement of NMT has been shown in several MT evaluation campaigns [27, 17, 119]. Moreover, fine-grained analysis (such as fluency, word reordering) [13] and translation quality [14], have shown large improvements for neural-based approaches over PB-SMT. Despite the progress, NMT performance is dependent on the amount and variety of available (parallel) training data, hence limiting

the usable application of NMT to high-resource pairs, such as; *English-French* and *English-German*. NMT difficulty in learning from a small number of examples have been discussed in [88], where performance degrades in comparison with PB-SMT. Recently, [142] have shown low-resource specific NMT model adaptation techniques that can improve over PB-SMT. Although, if a language pair is low-resourced, the training data is shallow, resulting in rare words; NMT fails to properly translate [88]. For the majority of the world's languages, these resources are not available. Not benefiting from high quality MT (as it is usually the case with high-resource languages) means that people's access to different sources of information can be restricted.

In Chapters 3, 4, and 5, we focus on the translation task of languages with small parallel data (low-resource) and languages only with monolingual data (zero-resource). Below, we give a summary of the proposed approaches.

2.6.1.1 Low-Resource NMT

We begin Chapter 3, by discussing the motivation behind addressing low-resource NMT performance, followed by a summary of the existing approaches. Several approaches have been suggested to address low-resource training conditions. A multilingual modeling approaches; with language-specific encoder and decoder [52], a language flag based single encoder-decoder [77, 64]. A transfer-learning from a high-resource to a low-resource model [187]. Both multilingual models and transfer-learning approaches have shown improvements for low-resource pairs. However, a thorough investigation involving more low-resource language pairs was not performed. In Sec. 3.1, we focus on investigating the benefit of a multilingual model for low-resource languages. We show how a single encoder-decoder multilingual model can outperform six different models trained on individual parallel corpora. We further compare a pivoting inference with the multilingual and single pair models. This will take us to further scale the number of the language directions in a single model and instead to attempt a direction zero-shot inference than pivoting.

The potential of a multilingual NMT for a direct zero-shot translation has been shown in [77, 64]. Relying on the simple but effective principle of source side language-flag, we present a 20 direction multilingual model for the first time as part of the IWSLT 2017 [24] evaluation campaign. In Sec. 3.2, we show multilingual NMT can scale and outperform

single pair NMT models with a minimal training complexity. Moreover, by only training only on 16 directions, 4 directions are investigated as a zero-shot setting. These models present an in-depth view of the potential behind multilinguality for low-resource language and zero-shot translation, which concludes with a call for further investigation.

Hence, in Sec. 3.3 of the chapter, we provide the result of our investigation comparing recurrence [70] and self-attention [168] based multilingual modeling. The comparison spans translation outputs from a single language pair, multilingual, and zero-shot translations. As metrics, we rely on an automatic (BLEU, TER) and human evaluation using multiple reference texts.

2.6.1.2 Zero-Shot NMT

The findings in Chapter 3 have solidified the possibility of improving the translation of low-resource languages leveraging a multilingual NMT approach. Moreover, zero-shot translation has shown a promising result without any additional modeling objective [77]. Hence, our work in Chapter 4, aim at improving zero-shot translation directions. Given a zero-shot translation direction we propose a self-learning approach from monolingual data of the source and target languages. Our approach is inspired by the semi-supervised learning principle of *back-translation* [136] and a language model based *dual-learning* from monolingual data to improve pre-trained NMT models [180].

Given, a multilingual model and a zero-shot directions (source $\leftrightarrow$ target) with the respective monolingual data, our approach to zero-shot translation is formulated in three step (*inference* $\rightarrow$ *training* $\rightarrow$ *inference*) operations,

i. Infer: using a pre-trained multilingual mode, generate translations in the *target** language from the source monolingual data.

ii. Train: the pre-trained multilingual model (with) the original parallel data and the newly generated *target** $\rightarrow$ *source* pair.

iii. Infer: using the target monolingual data, generate translations in the *source** language, for the next stage of training.

By simply iterating on the above three steps, for a certain number of rounds, our approach showed to progressively correct the generated outputs. Our findings show that

the proposed approach ultimately leads to − a self-learning zero-shot NMT modeling. We provide a comparison of the proposed zero-shot translation approach against a model trained with parallel examples is discussed in results and analysis part of Chapter 4.

2.6.2 Transfer-Learning for Low-Resource Languages

One of the key ingredients for improving NMT for a low-resource language pair is the level of shared linguistics characteristics with a high-resourced language pair. In simple terms, the more shared features a low-resource language pair exhibits with the high-resource pair, the better the chance of improvements. As such, a positive transfer of knowledge across and within NMT models is referred to as *transfer-learning*.

Transfer-learning can have at least two forms of application in NMT. The first is a model trained with a high-resource pair (*parent model*) is used to initialize a (*child*) model training with low-resource data [188]. The parent model can also be trained with multilingual data [113]. The other approach is training a single multilingual model with the aggregation of all the parallel pairs from low-resource and high-resource languages [77, 64]. In a multilingual model the transfer-learning can happen implicitly, based on the assumption that the high-resource pairs bring more diversity to training examples.

Both within and across a (multilingual) NMT transfer-learning approaches have demonstrated better performance improvement for the low-resource pairs. However, when transferring knowledge from the parent model, the child model parameters are fixated on the parent model parameters. Where the dependency enforces a one-size-fits-all approach, leading to a underperformance and unwanted training complexity since the parent models are mostly large with respect to the child. The exemplary aspect of the dependency on the parent model includes the word segmentation rules and the dictionary.

Hence, we propose a dynamic transfer-learning approach that addresses the full reliance on the pre-trained model parameters. The basic principle behind our approach is to update the pre-trained model vocabularies and the associated model parameters by first tailoring it to the low-resource language pair. In Sec. 5.2 of Chapter 5 we show how our proposal outperformed in a large margin a transfer-learning approach fixated on the parent model.

At time of transfer-learning from a parent to a child model, mixing the low-resource

language pair of the latter with a closely related language has given better improvements [113]. Mixing data from the closest language pair prevents over-fitting of the model on the low-resource pair by acting as a regularizer. Early findings have also showed transfer-learning is more effective if performed between related language pairs [114]. Moreover, the discussion in Chapters 3 and 4 shows that language relatedness not only helps to improve a pair with small parallel data but also a zero-shot translation task. In light of these findings, it is profound to hypothesize that relevant data selection from a pool of language pairs can better improve the transfer-learning for a low-resource language pair. This assumption is highly associated with the fact that a language can have multiple related languages ranked on the level of mutual intelligibility. However, for specific data at time of transfer-learning, the most relevant examples can come from different languages instead of only from the most related one. Thus, in addition to the dynamic transfer-learning approach, in Sec. 5.3 we propose to apply a language model perplexity based data selection strategies for maximizing the efficiency of the proposed transfer-learning approach.

2.6.3 Translation into Language Varieties and Styles

Until now we have seen the positive impact of language relatedness for improving low-resource and zero-resource NMT. We have highlighted that the degree of similarity between high-resource and low-resource languages can highly affect the expected improvement in the latter. In Chapters 6 and 7, we focus on more subtle translation requirements following the same principle behind multilingual NMT. In a similar way as to improve less-resourced languages translation, the demand for a more subtle and higher quality translation is rising. Some of these demands include the ability of a model to translate not only to a target language but to a specific dialect, or a short and concise translation style.

As such, the level of shared linguistics characteristics within a language variety and across languages can play a significant role to build a better model. Note that, in the real world setting all varieties and style in a given language does not have equal training data proportion. As a result, the data imbalance makes the problem of translating into varieties and styles, in part a low-resource task. These subtle similarities and differences for a variety (such as dialects) of the same language can occur in the level of vocabulary,

grammatical structure, and scripts. For styles: the similarity is more dominant than differences that are mostly restricted to a rephrasing of the same content in a different style of expression (such as formal or informal, short or long). Overall, modeling the translation task into language varieties and styles using a single model is advantageous to maximize the transfer-learning capability. Moreover, a single model approach also tends to maximize the ambiguities of generating the correct variety, a challenge we aim to address in our proposed solutions.

2.6.3.1 NMT into Language Varieties

Building separate NMT models for different varieties of the same language has been attempted in several works [131, 37, 38]. However, depending on the amount of available data in each variety, model performance is highly affected. For instance, data for the Portuguese-English pair exist on a large scale, however, if we split it into the European and Brazilian Portuguese, the latter will constitute much of the resource [163]. As a result, building separate NMT models will lead to a low-resource variant by splitting the available data. Moreover, a natural question is how to efficiently utilize parallel data for a given language pair without a variety-specific label. Our proposal in Chapter 6 shows how multilingual NMT can be extended to address the problem of translating into specific language varieties.

2.6.3.2 Controlling the Verbosity of NMT

While in Chapter 7, we focus on modeling a NMT to control the style of the generated output, more specifically the length of each translation. As in translating into language varieties, we first follow the same principle of classifying NMT training examples into different length criteria (such as short and long). Then, we utilize the length label for each segment to train a length style aware NMT model. Then, we show how injecting length information into the model at training time allows to control the conciseness level of the NMT output. Hence, the goal of our proposal both in language varieties and length style aware NMT modeling is to utilize all the data available within a single model, while, finding a way to control and disambiguate the generated outputs to the right target.

Chapter 3

Multilingual NMT for Low-Resource Languages

In this chapter, we first investigate the effectiveness of a single encoder and decoder-based multilingual modeling approach for improving low-resource language translation. Covering up to six language directions, we show how a low-resource multilingual setting can achieve large improvements over training language pair specific models. Then, by scaling the number of translation directions we explore the impact on zero-shot translation performance. Moreover, we probe pivoting translation as an alternative approach for achieving a zero-resource translation task. Finally, we focus on an empirical comparison of recurrence and transformer based seq2seq architectures on multilingual NMT (M-NMT). We provide quantitative analysis on bilingual, multilingual and zero-shot translation outputs. By leveraging multiple post-edits of automatic translations we demonstrate how language relatedness influences zero-shot translation.

In the rest of the chapter, we begin in Sec. 3.1 with a summary of the problem statement and the motivation for improving low-resource language translation with a multilingual modeling approach. In Sec. 3.2, we expand our discussion of M-NMT focusing on zero-shot translation. Following the experimental results, we analyze the performance of the zero-shot and pivoting translation mechanisms against single language pair models. Finally, Sec. 3.3 is dedicated to a systematic comparison of recurrent and transformer models. We discuss related work, summarize evaluation criteria, and provide an in-depth analysis of the translation outputs from a supervised, zero-shot and multilingual systems.

3.1 Multilingual Model for Low-Resource Languages

3.1.1 Problem Statement and Motivation

NMT has recently shown its effectiveness by delivering the best performance in various evaluation campaigns (IWSLT 2016 [27], WMT 2016 [17]). Unlike rule-based or phrase-based MT, the end-to-end learning approach of NMT models the mapping from source to target language directly through a posterior probability. Despite the continuous improvement in performance and translation quality, NMT models are highly dependent on the availability of large parallel data, which in practice can only be acquired for a very limited number of language pairs. For this reason, building effective NMT systems for low-resourced languages becomes a primary challenge [88].

To address the challenges in low-resource translation task, we focus on M-NMT [77, 64] modeling, which allows to train multiple translation directions in a single model. Our motivation is that positive cross-lingual transfer [153] via parameter sharing should ideally help in the case of similar languages and sparse training data. Hence, we investigate multilingual modeling across Italian, Romanian, and English languages, and simulate low-resource conditions by limiting the amount of parallel data.

3.1.2 Related Work

Multilingual NMT aims to model translation across multiple languages, based on the end-to-end training approach in NMT. Early works in M-NMT are characterized by the use of separate encoder, decoder, and an attention mechanism for every language direction [43, 102]. For the first time [52] introduced a way to share the attention mechanism in a many-to-many translation setting while keeping separate encoder and decoder networks for each source and target language.

Moreover, most of the initial approaches to M-NMT required modifications on the standard encoder-decoder architecture [186, 52, 53, 43, 102, 99]. State-of-the-art results are achieved by simply decorating an NMT inputs with special language tags, to direct the model to a preferred target language at inference time [64, 77]. The artificial language tag (also known as *target-forcing, language-flag*) is prepended at a preprocessing stage to

the source sentences in order to enable multilingual translation. More specifically, the approach in [64] appended a language-specific code to differentiate words from different languages. This word and sub-word level language-specific coding mechanism is proved to be expensive, by creating longer sentences that can deteriorate the performance of NMT [30]. In [77], however, only one artificial token is prepended at the beginning of the source sentences. This single token, which specifies the target language proved to work with a comparable performance as in [64].

3.1.3 Multilingual NMT for Low-Resource Languages

As discussed in Chapter 1, a multilingual translation task can be categorized into many-to-one, one-to-many, or many-to-many directions, with an increasing difficulty. In [64, 77] it has been shown that a multilingual system trained on a large amount of data improves over a baseline bilingual model, and it is also capable of performing zero-shot translation. Moreover, by employing one of these scenarios, recent works in M-NMT have shown the possibility of translating across language pairs never seen at training time [64, 77]. For our experimental setting, we follow the many-to-many M-NMT scenario using the approach in [77].

Our goal is to show that it is possible to train a single NMT model for the translation task between multiple language pairs in a low-resource setting. Hence, we focus on M-NMT in a resource-scarce setting and show how M-NMT is never worse than a bilingual system for each of the language directions used in the training phase. In fact, the multilinguality can be considered as a way to increase the available amount of data for language directions with small data.

Moreover, only a single system is needed with respect to several bidirectional NMT systems, thus our setting also represents a way for saving training time and compresses the number of required parameters. The target language can be imposed on the network by using the previously described target-forcing mechanism. Furthermore, we use our multilingual model to perform zero-shot translation, for the language pairs without parallel data as in [77]. We hope that by simply applying the target-forcing in the zero-shot scenario, the system can generate sentences in the target language.

Pivoting Translation is a rather intuitive way to approach zero-shot translation,

especially when it involves low-resourced languages. The idea is to translate from/into under-resourced languages (L_{source} and L_{target}) by leveraging data available for a high-resourced one (L_{pivot}) used as "bridge" between the two languages (*i.e.* $L_{source} \rightarrow L_{pivot} \rightarrow L_{target}$) [177]. However, results in the pivoting framework are strictly bounded to the performance of the two combined translation engines, and especially to that of the weaker one. In contrast, multilingual models that leverage knowledge acquired from data for different language combinations (similar to multi-task learning) can potentially compete or even outperform the pivoting ones. We expect to achieve a comparable pivoting results using a single multilingual model.

3.1.4 Experiments

Our NMT model uses embeddings with dimension 1024 and RNN layers based on GRUs of the same dimension. The optimization algorithm is Adagrad [44] with an initial learning rate of 0.01 and mini-batches of size 100. Dropouts are used on every layer, with probability 0.2 on the embeddings and the hidden layers and 0.1 on the input and output layers. All experiments are done using the NMT toolkit Nematus [135].[1]

Language pair	**Train**	**Dev10**	**Test10**	**Test17**
English - Italian	231619	1643	929	1147
English - Romanian	220538	1678	929	1129
Italian - Romanian	217551	1643	914	1127

Table 3.1: *Parallel segments used for training and evaluation in a low-resource scenario.*

For the training set, we used the dataset provided by the IWSLT2017 multilingual shared task for all possible language pair combinations between Italian (It), Romanian (Ro) and English (En) [25].[2] At the preprocessing stage, we applied word segmentation by jointly learning the Byte-Pair Encoding [140], merging rules set to 39,500. The size of the vocabulary both in case of the bilingual and the multilingual models stays just under 40,000 sub-words. An evaluation script to determine the BLEU [121] score is used to validate on the dev set and later to choose the best performing models.

[1]Nematus: https://github.com/EdinburghNLP/nematus

[2]International Workshop on Spoken Language Translation: http://workshop2017.iwslt.org/

Direction	**NMT**	**M-NMT**
English → Italian	26.79	26.34
Italian → English	31.43	31.39
English → Romanian	21.55	**22.13**
Romanian → English	33.84	34.16
Italian → Romanian	15.60	15.92
Romanian → Italian	21.00	**21.60**

Table 3.2: *Comparison between six bilingual models (NMT) against a single multilingual (M-NMT) model. A difference of* ≥ 0.5 *BLEU is bold highlighted.*

We trained models for two different scenarios, the first is the multilingual scenario containing all the available language pairs, while the second scenario is the zero-shot using pivoting, which does not contain parallel sentences for the Romanian ↔ Italian language pairs. For development and evaluating the models, we used sets from the IWSLT 2010 [123] and IWSLT2017 evaluation campaign. The inference is performed using beam search of size 12.

3.1.5 Results

We discuss the experimental results, first comparing the single pair NMT models with a multilingual model, then we evaluate multilingual pivoting strategy for pairs with no parallel data.

3.1.5.1 Bilingual Vs. Multilingual

In the first scenario, we compare the translation performance of independently trained bilingual models against the M-NMT model. In total there are six bilingual models, whereas the M-NMT is trained using the concatenation of all the six languages pair dataset, by just appending an artificial token on the source side. As shown in Table 3.2 and 3.3, the performance of our systems are evaluated on dev2010 and test2017.

Our preliminary experiments show that the M-NMT system favorably compares with

Direction	**NMT**	**M-NMT**
English → Italian	27.44	**28.22**
Italian → English	29.9	**31.84**
English → Romanian	20.96	**21.56**
Romanian → English	25.44	**27.24**
Italian → Romanian	17.7	**18.95**
Romanian → Italian	19.99	**20.72**

Table 3.3: *Comparison between six bilingual models (NMT) against a single multilingual (M-NMT) model on test2017.*

the bilingual systems. Improvements are observed in several language directions, which are likely gained from the cross-lingual parameter transfer between the additional language pairs involved in the source and target side. Specifically, the M-NMT showed an improvement of +0.58 and +0.60 for En→Ro and It→Ro directions, while having only a small decrease in performance for the En→It and It→En directions (see Table 3.2).

For the evaluation using test2017, however, the M-NMT performed better in all directions than the NMT models (see Table 3.3). These results show that the M-NMT model performs either in a comparable way or outperforms the single language pair models in this resource-scarce scenario. Moreover, the simplicity of using a single model instead of six leaves a room for further improvements by incorporating more language pairs.

3.1.5.2 Pivoting using a Multilingual Model

The pivoting experiment is set up by dropping the Italian-Romanian language pairs from the six directions M-NMT model, which gives us a four directions multilingual model (we call it, PM-NMT), where all the configurations stay the same as in M-NMT. As demonstrated by Johnson et al. [77], our first option is to attempt a direct zero-shot translation both for Italian → Romanian and Romanian → Italian directions using the PM-NMT model. However, our investigation of a direct zero-shot inference using the PM-NMT model consistently resulted in a performance below our expectation (i.e., < 1 BLEU score). Hence, we opted for a pivoting mechanism to enable the zero-resource

Direction	P-NMT	PM-NMT	Δ BLEU
Italian → Romanian	14.14	14.75	+0.61
Romanian → Italian	20.16	19.72	−0.44

Table 3.4: *Comparison of pivoting with two bilingual models (P-NMT) against pivoting using one multilingual model (PM-NMT). Both approaches use English as the pivoting language.*

Direction	P-NMT	PM-NMT	Δ BLEU
Italian → Romanian	16.3	17.58	+1.28
Romanian → Italian	18.69	18.66	−0.03

Table 3.5: *Comparison of pivoting with two bilingual models (P-NMT) against pivoting using one multilingual model (PM-NMT) using test2017 as the evaluation set.*

translation task of the Italian ↔ Romanian pair. In pivoting translation, our main aim is to analyze how a multilingual model can improve the zero-resource task in comparison with utilizing two bilingual models (P-NMT). For our experiment, we use English as the bridge/pivot language.

The results in Table 3.4, show the potential, although partial, of using multilingual models with pivoting for unseen translation directions. The comparable results achieved in both directions speak to us in favor of training and deploying one M-NMT system instead of two distinct NMT.

From the evaluation results on *test2017*, we confirmed that M-NMT can achieve a comparable (Ro → It) or better (It→Ro) result over the two NMT systems used for pivoting.

3.1.6 Summary

In this section, we used a M-NMT model in a low-resource language pairs scenario. We showed that a single multilingual system achieves a comparable performance with the bilingual baselines while avoiding the need to train several single language pair models.

Then, we showed how a multilingual model can be used for zero-shot translation by using a pivot language for achieving slightly lower results than a bilingual model trained on that language pair.

These results are important indicators for understanding the weakness and strength of multilingual modeling, particularly, in the low-resource and zero-resource settings. Possible next directions could be: to explore how the choice of different languages and number of directions can enable a better parameter transfer in a single model, achieving a direct zero-shot translation in zero-resource scenario instead of relying on the pivot language (e.g. *English*), and exploring alternative seq2seq modeling architectures.

In the next section of this chapter we ask and investigate the importance of scaling multilingual translation directions and ultimately aim at achieving a competitive direct zero-shot translation, instead of relying on pivoting translation.

3.2 Scaling Multilingual Translation Models

3.2.1 Background

Multilingual NMT has been shown to facilitate cross-lingual knowledge transfer and ultimately enabling zero-shot translation between language pairs never seen at training time. Despite promising results, multilingual model can fail to achieve a reasonable zero-shot translation performance. In the previous section, we have demonstrated poor zero-shot performance of a multilingual model in a particular low-resource setting. One of the reasons for this shortcoming could be the dependency of the *target-forcing* mechanism on a large scale training examples as in [77], to effectively handle zero-shot translation. This is particularly visible in case of a zero-shot target language which appears only once in comparison with other source $\rightarrow$ target pairs in the multilingual model data. Hence, the zero-shot result using the multilingual modeling in [77, 64] requires further investigation, to verify if the method can work in various language settings, particularly for low resourced and across distant languages.

Hence, focusing on low-resource multilingual setting, in this section we evaluates two multilingual systems that are trained on five languages (*English, Dutch, German, Italian, and Romanian*). The first one is a 20 language direction model, which handles all possible combinations of the five languages. While the second system is trained only on 16 directions, leaving the others as zero-shot translation (*i.e.,* representing a more complex inference task on language pairs not seen at training time). Specifically, the zero-shot directions are Dutch $\leftrightarrow$ German and Italian $\leftrightarrow$ Romanian (resulting in four language combinations).

We compare and show the results of the two multilingual models against a baseline single language pair systems. Particularly, we focus on the four zero-shot directions and evaluate how a multilingual model trained with small data can provide reasonable results. Furthermore, we show how pivoting (*i.e.* using a bridge or pivot language for inference in a source $\rightarrow$ pivot $\rightarrow$ target translations) using a multilingual model can be an alternative to enable zero-resource translation between the source and target language.

In the following sections, we present the findings from the two multilingual models. For convenience, we refer to the 20 direction model as *M-NMT* and the 16 direction as

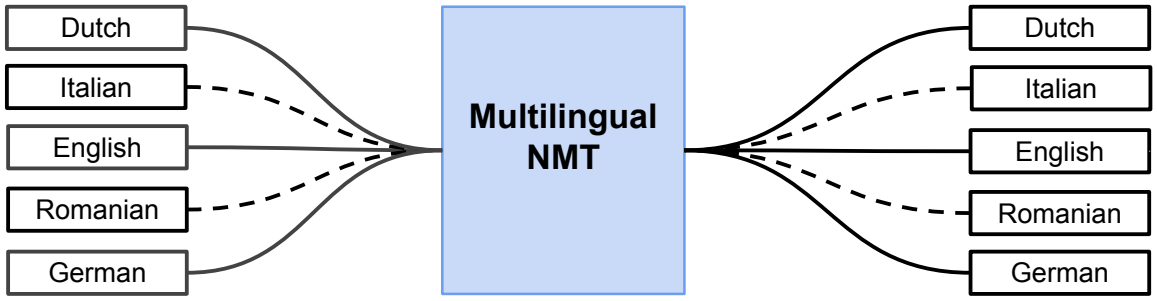

Figure 3.1: *The multilingual system source → target association. Parallel data exist for all the* 20 *directions in the first multilingual model, whereas for the zero-shot model the Dutch ↔ German and Italian ↔ Romanian pairs (dashed line) are excluded.*

zero-shot *Z-NMT* models.[3] We trained the two models separately, by sharing a common configuration. The only difference, at training time, is that we removed the four language directions involved in the zero-shot task. Figure 3.1, illustrates the twenty possible associations between the source and target pairs, avoiding (*source=target*) condition. We trained the models following the same preprocessing and training procedures described in [77]. Note that, due to its small size ($\approx 200K$ for each language pair), the training data set becomes even more sparse after preprocessing and dropping sentences above a certain length (which becomes necessary to facilitate and speed-up the training process).

3.2.2 Experimental Settings

3.2.2.1 Training Details

For training the multilingual and the single language pair systems we used a standard encoder-decoder NMT architecture with attention mechanism [104, 151]. The encoder and decoder sides of the network consist of four layers, where the first two layers of the encoder are bidirectional. Table 3.6, shows the parameters used for training both the multilingual and single language pair systems. For optimization, we used Adam [82] with a learning rate of 0.001. Learning rate decay of 0.5 is applied if the perplexity does not decrease on the validation set or the number of epoch passes 8. For reducing perplexity

[3]Models are also used for IWSLT2017 [24] shared task participation's: *i)* a multilingual translation task in a small data condition for 20 language directions, and *ii)* a multilingual zero-shot task in a similar small data condition [93].

Model Parameters	**Value**
RNN type	LSTM
RNN size	1024
embedding	512
encoder	bidirectional
encoder depth	4
decoder depth	4
optimizer	adam

Table 3.6: *Parameters used to train both single language pair and multilingual models.*

and the network size, we also share the word and softmax embedding of the decoder as suggested by Press and Wolf [126]. To prevent overfitting [150], particularly for the training dataset in this low-resource setting, we applied a dropout of 0.3 on all layers [55]. At time of inference, a beam search of size 10 is utilized to balance decoding time and accuracy of the search. Where each decoding step takes a batch of 128 evaluation set. The experiments are carried out using the open source OpenNMT-py toolkit [83].[4]

We trained the twenty single-language pair models with the same amount of training data used by each direction of the multilingual models (see Table 4.2 for details). For every direction of the multilingual models and every single pair model we report case sensitive detokenized (*i.e.,* using the internal tokenization of the scorer) BLEU scores [121] computed using mteval-v13a.pl.

3.2.2.2 Data and Preprocessing

In the source-target pair of the five languages considered in this work, there are $\approx 200k$ parallel training sentences in each pair. As shown in Table 3.7, test2010 is used for evaluating the models, whereas test2017 is used for comparison purposes and as the official submission test set.

To prepare the data for training, we first prepare a tokenized version. Then, using a shared byte pair encoding (BPE) model, we segment the tokens into sub-word units [137].

[4]OpenNMT: https://github.com/OpenNMT/OpenNMT-py

Language Directions	Train	Test 2010	Test 2017
English ↔ German	197,489	1,497	1,138
English ↔ Italian	221,688	1,501	1,147
English ↔ Dutch	231,669	1,726	1,181
English ↔ Romanian	211,508	1,633	1,129
German ↔ Italian	197,461	1,502	1,133
German ↔ Romanian	194,257	1,626	1,121
Dutch ↔ Italian	228,534	1,623	1,183
Dutch ↔ Romanian	199,762	1,637	1,123
German ↔ Dutch	209,169	1,729	1,174
Italian ↔ Romanian	209,668	1,605	1,127

Table 3.7: *Number of sentences used for training and evaluation. The German ↔ Dutch and Italian ↔ Romanian four language directions are removed from the training data of the zero-shot multilingual model.*

The BPE model is trained on a joint source and target dataset covering all the language directions. For this operation we used 8,000 BPE merging rules. A frequency threshold of 30 is used to apply the segmentation. For choosing the BPE segmentation rules, we follow the suggestion in [40] in such small data condition. When training the multilingual models, we add the *target-forcing* language token at the source side of each parallel data, both for training and validation sets [77].

3.2.3 Results

3.2.3.1 Single Language Pair Models

As discussed in training details, these models are trained in a similar setting with the multilingual models. Table 3.8, summarizes the performance each of the twenty models on *test2017*. Except for the slight gain in the Romanian → Italian direction over the results of the multilingual model (see Table 3.9), the performance of the single language pair models (see Table 3.8), are poorer in the rest of the other 19 directions.

S-NMT	En−De	En−Nl	En−It	En−Ro	De−Nl	De−It	De−Ro	Nl−It	Nl−Ro	It−Ro
→	19.84	26.41	29.90	21.41	18.93	15.52	12.52	18.47	14.71	18.67
←	24.69	30	34.03	28.03	17.93	15.47	13.81	20.13	16.78	**21.71**

Table 3.8: *BLEU score on IWSLT tst2017 from twenty single language pair models. The Romanian → Italian direction is the only gain over the multilingual system.*

3.2.3.2 Multilingual Models

In this experiment, we present the multilingual (M-NMT) 20 direction and zero-shot (Z-NMT) 16 direction models. Note: in case of the zero-shot model the training data for the German ↔ Dutch and Italian ↔ Romanian directions are dropped. As in the single language pair models, the rest of the training follows the procedures given in training details. The results shown in Table 3.9, are the primary runs of the official submission for the multilingual and zero-shot small data condition tasks. The term of comparison between these two multilingual models is focused on the four zero-shot directions. As expected, the zero-shot model performed poorer than the multilingual model in all of the four directions.

M-NMT	En−De	En−Nl	En−It	En−Ro	De−Nl	De−It	De−Ro	Nl−It	Nl−Ro	It−Ro
→	20.88	26.72	29.6	21.95	19.16	16.84	14.62	19.33	16.54	19.06
←	25.62	29.79	34.24	28.93	18.59	16.88	15.87	20.27	18.92	21.34
Z-NMT										
→	20.67	26.11	28.86	21.54	17.17	16.28	13.93	**19.76**	15.88	16.58
←	25.22	**30.04**	34.16	28.52	16.96	16.13	15.47	20.00	17.72	18.32

Table 3.9: *BLEU for the IWSLT tst2017 using the multilingual (M-NMT) model trained on* 20 *directions and the zero-shot model (Z-NMT) trained using parallel data from* 16 *directions. Bold highlighted Nl → En and Nl → It are the only cases where the zero-shot model performed better than the multilingual.*

Particularly, we see a larger gap of 3.02 for the Romanian → Italian, whereas the Italian → Romanian direction has a difference of 2.48 BLEU score. In case of German → Dutch and Dutch → German the gap closes to 1.99 and 1.63 respectively. For the other 16 non-zero-shot directions, the multilingual model performed slightly better than the

zero-shot model. However, in case of Dutch → English and Italian→Dutch there exists a pattern where the zero-shot model performed better.

3.2.3.3 Zero-shot Vs. Pivoting Translation

Approaches	De → Nl	Nl → De	It → Ro	Ro → It
Zero-shot	17.17	**16.96**	16.58	18.32
Zero-shot Pivot	**17.67**	16.84	**17.3**	**19.57**
Single Pair Pivot	15.3	14.9	15.22	17.2

Table 3.10: *BLEU score comparison of German ↔ Dutch and Italian ↔ Romanian four language directions using three different zero-shot translation mechanisms. The first row is a direct zero-shot translation using the Zero-shot model, while the last two rows show the results of the pivoting mechanism.*

Furthermore, we compare zero-shot translation mechanisms using the zero-shot multilingual model and models trained in a single language pair setting. Specifically, we compared three different results of a zero-shot translation on the IWSLT *tst2017*. The first is a direct zero-shot from a source → target language using the Zero-shot multilingual model. The other two results are acquired through a pivoting translation mechanism in a *two-step* translation. Hence, pivoting using single language pair models requires a source → pivot and a pivot→target model. However, this is not the case for the Zero-shot model which assumes to already have the pivot paired with the source and target languages. In both cases, we use English as a pivot language. Thus, for the Italian ↔ Romanian zero-shot directions we follow Italian ↔ English ↔ Romanian, whereas the German ↔ Dutch translation is done as German ↔ English ↔ Dutch *two-step* translations.

The results in Table 3.10, shows better performance of the Zero-shot model using a pivoting mechanism (except the Nl → De direction). In a surprising way, the pivoting using two separate single language pair models for each translation direction perform worse than the direct zero-shot and the pivoting zero-shot using the multilingual model in row 1 and 2.

3.2.4 Summary

The experimental results show that a single multilingual system can perform better than independently trained single language pair systems. Hence, training a single system on the concatenation of all the language directions helps to maximize the parameter sharing from the common representation space. Unlike the scenario in previous work [77], we showed the improvements in a low resource setting, without any additional data to tune the system. In the case of the zero-shot model, we considered the non-zero-shot 16 directions for comparison with the bilingual models. In a similar way with the multilingual model, the zero-shot model has shown gains over the single language pair models.

Even though the zero-shot model showed a comparable performance with the multilingual model in the 16 non-zero-shot directions, there is a slight performance degradation in all but the Dutch→English direction. For instance, a 29.6 BLEU score for English→Italian of the multilingual model decreases to 28.86 BLEU with the zero-shot model. However, for the translation directions Source→English the maximum loss for the zero-shot model is 0.41 BLEU in the Romanian→English direction. As we expected initially, these results reflect a condition where the number of language pairs with English (on the encoder and decoder side) stayed the same in both multilingual models. In contrast the absence of the four zero-shot (source↔target) combinations influenced the translation performance of the zero-shot model even for the language pairs seen at training time.

The pivoting strategy is another way of showing the reasonable performance of the zero-shot model, except it relies on a third language, in our case English. The two-step inference (i.e., source → pivot, and then pivot→target) provided a better performance in three directions out of four (see Table 3.10), in comparison with a direct zero-shot translation. We observed that using English (the only language that has a pair and better performance with all the zero-shot directions) as the bridge language played a major role for the gain. However, as discussed in the background, pivoting using two separate bilingual systems is found to be weaker (see the third row of Table 3.10) in leveraging the pivot language. This can be observed from the weaker bilingual systems in comparison with the zero-shot model, see Table 3.8 and 3.9.

Overall the reasonable performance of the zero-shot model confirms the potential of a multilingual approach as in [77]. Note that the key difference with the experimental

setting in Sec. 3.1, is that we have introduced more translation directions, even if it is still a low-resource setting. Hence, by simply maximizing the language pairs in a single model, there is a higher chance of achieving better zero-shot translation. Moreover, for achieving a better zero-shot translation we expect that finding the right language combinations, amount of data, and the number of languages requires further investigation. Furthermore, a human evaluation on the outputs of the bilingual and the multilingual models would be interesting to assess the translation quality, in addition to confirming the evaluation scores, reported in this section. Thus, in the following section, we compare and systematically analyze the translation outputs of zero-shot and non-zero-shot systems, and compare the performance of recurrent and transformer based seq2seq multilingual modeling approaches.

3.3 Recurrence Vs. Self-Attention on Multilingual NMT

As we have discussed and demonstrated in the previous sections (Sec. 3.1 and 3.2), M-NMT showed competitive performance against bilingual systems. Notably, in low-resource settings, it proved to work effectively leveraging the shared representation space that is forced across languages and induces a sort of transfer-learning. Furthermore, M-NMT enables zero-shot inference across language pairs never seen at training time. Despite the increasing interest in this framework, an in-depth analysis of what a M-NMT model is capable of and what it is not is still missing.

This section, *i*) provides a quantitative and comparative analysis of the translations produced by bilingual, multilingual and zero-shot systems; *ii*) investigates the translation quality of two of the currently dominant neural architectures in MT, which are the recurrent and the transformer ones; and *iii*) quantitatively explores how the closeness between languages influences the zero-shot translation. Our analysis leverages multiple professional post-edits of automatic translations by several different systems and focuses both on automatic standard metrics (BLEU and TER) and on widely used error categories, which are lexical, morphology, and word order errors.

In the following sections, we begin with a brief discussion on the background and motivation, followed by a review of related work on quantitative analysis of MT outputs. Then, we give an overview of NMT models contrasting recurrent and transformer approaches. In the experimental settings, we describe the dataset and preprocessing pipeline, qualitative evaluation data, training settings, models and the evaluation methods. For the qualitative analysis, we categorize our discussion into two: a general translation analysis, and a fine-grained analysis on lexical, morphological and word-order error types of the translation output. For the sake of a more clearer interpretation, both analysis types are further categorized in *related* and *unrelated* language settings.

3.3.1 Background and Motivation

As witnessed by recent machine translation evaluation campaigns (IWSLT 2017 [24], WMT 2017 [119]), in the past few years several model variants and training procedures

have been proposed and tested in NMT. Models were mostly employed in conventional single language-pair settings, where the training process exploits a parallel corpus from a source language to a target language, and the inference involves only those two languages in the same direction. However, there have also been attempts to incorporate multiple languages in the source [102, 186, 99], in the target [43], or in both sides like [52] which combines a shared attention mechanism and multiple encoder-decoder layers. Regardless, the simple approach proposed in [77, 64] remains outstandingly effective: it relies on single "universal" encoder, decoder and attention modules, and manages multilinguality by introducing an artificial token at the beginning of the input sentence to specify the requested target language.

Besides specific studies focusing on new architectures and modules, like [104] that empirically evaluates different implementations of the attention mechanism, the comprehension of what a model can learn and the errors it makes has been drawing much attention of the research community, as evidenced by the number of recent publications aiming at comparing the behavior of neural vs. phrase-based systems [13, 166, 14]. However, understanding the capability of M-NMT models in general and zero-shot translation, in particular, has not been thoroughly analyzed yet. By taking the bilingual model as the reference, this work quantitatively analyzes the translation outputs of multilingual and zero-shot models, aiming at answering the following research questions;

- How do bilingual, multilingual, and zero-shot systems compare in terms of general translation quality? Is there any translation aspect better modeled by each specific system?
- How do Recurrent and Transformer architectures compare in terms of general translation quality? Is there any translation aspect better modeled by each specific system?
- What is the impact of using related languages data in training a zero-shot translation system for a given language pair?

To address these questions, we utilize the data collected in the IWSLT 2017 MT evaluation campaign [24], featuring five languages (*English, Dutch, German, Italian, and Romanian*) and all their twenty possible translation directions. In addition to the official

external single reference of the test sets, we can also rely on professional post-edits of the outputs of nine Romanian → Italian and of nine Dutch → German participants' systems. Hence, we exploit the availability of multiple Italian and German references to perform a thorough analysis for identifying, comparing and understanding the errors made by different neural system/architectures we are interested in; in particular, we consider pairs of both related languages (Romanian → Italian, Dutch →German) and unrelated languages (Romanian → German and Dutch → Italian). Furthermore, to explore the impact of using data from other related languages, French and Spanish are considered for training purposes as well, in particular for analyzing the behavior of zero-shot *Source* → Italian systems, *Source* representing any source language distant from Italian.

3.3.2 Related Work

Recent trends in NMT evaluation show that post-editing helps to identify and address the weakness of systems [14]. Furthermore, the use of multiple post-edits in addition to the manual reference is gaining more and more ground [13, 88, 166, 14]. For our investigation, we follow the error analysis approach defined in [14], where multiple post-edits are exploited in order to quantify *morphological, lexical, and word order errors*, a simplified error classification with respect to that proposed in [171], which settles two additional classes, namely missing and extra words.

The first work that compares bilingual, multilingual, and zero-shot systems comes from the IWSLT 2017 evaluation campaign [24]. The authors analyze the outputs of several systems through two human evaluation methods: *direct assessment* which focuses on the generic assessment of overall translation quality, and *post-editing* which directly measures the utility of a given MT output to translators. Post-edits are also utilized to run a fine-grained analysis of errors made by the systems. The main findings are that: *i*) a single multilingual system is an effective alternative to a bunch of bilingual systems, and that *ii*) zero-shot translation is a viable solution even in low-resource settings.

Motivated by the findings in [24] in this section we explore in more detail the practical feasibility of multilingual and zero-shot approaches. In particular, we explore the benefit of adding training data involving related languages in a zero-shot setting and, in that framework, we compare the behavior of state-of-the-art recurrent and transformer NMT.

3.3.3 Summary of NMT Architectures

A standard state-of-the-art NMT system comprises of an encoder, a decoder and an attention mechanism, which are all trained with maximum likelihood in an end-to-end fashion [9]. Although there are different variants of the encoder-attention-decoder based approach, this work focuses on the recurrent [151] and the transformer variants [168].

In both the recurrent and transformer approaches, the encoder is purposed to cipher a source sentence into hidden state vectors, whereas the decoder uses the last representation of the encoder to predict symbols in the target language. In a broad sense, the attention mechanism improves the prediction process by deciding which portion of the source sentence to emphasize at a time [104]. To support our discussion, we recall the two architecture types discussed in Sec. 2.2.2 and 2.3.3 of Chapter 1, and provide additional details in a comparative way.

3.3.3.1 Recurrent NMT

As discussed in Sec. 2.2.2, RNN to build the internal representations of both the encoder and decoder. While RNNs are in theory the most expressive type of neural networks [147], they are in practice hard and slow to train. In particular, the combination of two levels of deepness, horizontal along time and vertical across the layers, makes gradient-based optimization of deep RNNs slow to converge and difficult to parallelize [178]. Recent work succeeded in speeding up training convergence [134] of recurrent NMT by reducing the network size via parameter tying and layer normalization. On the other hand, the simple recurrent NMT model proposed by [41], which weakens the network time dependencies, has shown to outperform LSTM-based NMT both in training speed and performance.

3.3.3.2 Transformer NMT

The Transformer architecture works by relying on a self-attention (*intra-attention*) mechanism, removing all the recurrent operations that are found in RNN. In other words, the attention mechanism is repurposed to compute the latent space representation of both the input and output sequences. As summarized in Sec 2.3.3, both the encoder and decoder of a transformer based NMT model is composed of uniform layers, each built of

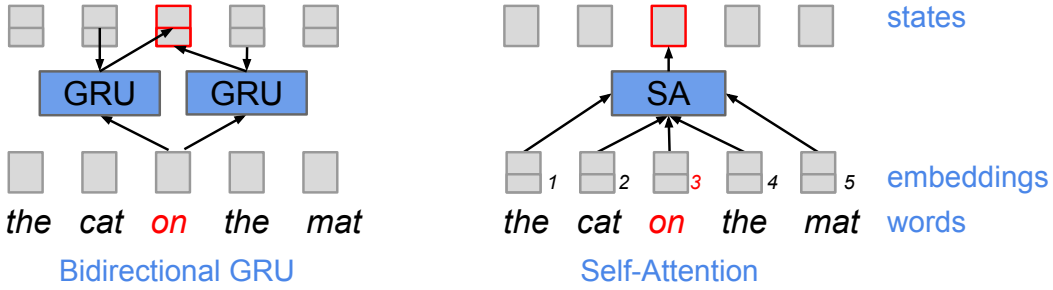

Figure 3.2: *Single-layer encoders with recurrent (left) and transformer networks (right). A bi-directional recurrent encoder generates the state for word "on" with two GRU units. Notice that states must be generated sequentially. The transformer generates the state of word "on" with a self-attention model that looks at all the input embeddings, which are extended with position information. Notice that all the states can be generated independently.*

two sub-layers, i.e., a multi-head self-attention layer and a position wise FNN layer. The multi-head sub-layer enables the use of multiple attention functions with a similar cost of utilizing attention, while the FNN sub-layer is a fully connected network used to process the attention sublayers; as such, FNN applies two linear transformations on each position and a ReLU [168].

The right-hand side of Figure 3.2 depicts a simple one-layer encoder based on self-attention. Notice that, in absence of recurrence, a *positional-encoding* is added to the input and output embeddings. Similarly, as the time-step in RNN, the positional information provides the transformer network with the order of input and output sequences. In our experiments, we use absolute positional encoding but, very recently, the use of relative positional information has been shown to improve the network performance [144].

Recall, in the recurrence based approach discussed in Sec. 2.2.2, learning long-range dependencies is not only expensive in terms of computational cost but creates a weaker signal at time of back-propagation, particularly for longer sequences [69]. The self-attention hidden representation, however, becomes advantageous since the distance between two input positions can be kept minimal. Vaswani et al. [168], compared the ability of self-attention based seq2seq network computational complexity (layer-wise) and the number of computation that can be parallelized. For a sequence of input tokens X with a max length L_x, self-attention has a complexity of $O(1)$, while a recurrent network requires

$O(L_x)$ operations. For a model dimension of d, layer-wise self-attention requires the lowest complexity of $O(d \cdot L_x^2)$, with respect to recurrence layers $O(d^2 \cdot L_x)$. Note that the optimal layer complexity of self-attention hold if $L_x < d$. Even if $L_x > d$, a different suggestion known as restricted *self-attention*, that considers only the closest r tokens in the sequence for the current output, this means the computation can be reduced to $O(r \cdot L_x \cdot d)$, and the maximum path length grows to $O(L_x/r)$ from $O(1)$.

The summarized characteristics of recurrent and transformer networks indicate that the latter has minimal computational complexity. Moreover, from the experimental results in Sec. 3.1 and 3.2 we have witnessed a better performance from the transformer approach. In the subsequent experimental and results section, we further investigate how these two architectures are compared in terms of both general and fine-grained translation performance.

3.3.4 Experimental Settings

3.3.4.1 Dataset and Preprocessing

The experimental setting comprises seven languages (English (En), German (De), Dutch (Nl), French (Fr), Italian (It), Romanian (Ro) and Spanish (Es); for each language pair, we use the ≈200,000 parallel sentences from IWSLT evaluation campaign [24]. Table 3.11, summarizes the amount of training, dev and test sentence pairs.

Language	Train	Dev	Test	Language	Train	Dev	Test
En - De	197,489	1,497	1,138	De - It	197,461	1,502	1,133
En - It	221,688	1,501	1,147	De - Ro	194,257	1,626	1,121
En - Nl	231,669	1,726	1,181	De - Nl	209,169	1,729	1,174
En - Ro	211,508	1,633	1,129	Nl - It	228,534	1,623	1,183
En - Fr	232,813	1,568		Nl - Ro	199,762	1,637	1,123
En - Es	219,464	1,564		It - Ro	209,668	1,605	1,127

Table 3.11: *Number of sentences used for training, development (test2010), and test (test2017) sets after a preprocessing step.*

In the preprocessing pipeline, the raw data is first tokenized and cleaned by removing

empty lines. Then, a shared byte pair encoding (BPE) model [137] is trained using the union of the source and target sides of the training data. The number of BPE segmentation rules is set to $8,000$, following the suggestion of [40] for experiments in small training data condition. For training transformer, the internal sub-word segmentation provided by the Tensor2Tensor library is used.Note that prepending the "language-flag" on the source side of the corpus is specific to the multilingual and zero-shot models.

3.3.4.2 Evaluation Data

For our investigation, we exploit the nine post-edits available from the IWSLT 2017 evaluation campaign. Post-editing regarded the bilingual, multilingual, and zero-shot runs of three different participants to the two tasks Nl $\rightarrow$ De and Ro $\rightarrow$ It. Human evaluation was performed on a subset (603 sentences) of the nine runs, involving professional translators. Details on data preparation and the post-editing task can be found in [24].

The translation directions we consider are Nl/Ro $\rightarrow$ De and Nl/Ro $\rightarrow$ It. The choice of *German* and *Italian* as the target languages is motivated by *i*) the availability of multiple post-edits for the fine-grained analysis and *ii*) the possibility of varying the linguistic distance between the source and the target languages, allowing experimental configurations suitable to answer the research questions raised in our motivation.

As said, for Nl $\rightarrow$ De and Ro $\rightarrow$ It the human evaluation sets consist of 603 segments. Since post-editing involved only those two language pairs, for the other two directions considered, namely Nl $\rightarrow$ It and Ro $\rightarrow$ De, we attempted to exploit at best the available post-edits by looking for all and only the segment pairs of the Nl $\rightarrow$ It and Ro $\rightarrow$ De tst2017 sets for which the target side exactly matches (at least) one of the segment pairs of the Ro $\rightarrow$ It and Nl $\rightarrow$ De human evaluation sets. This way, we were able to find 478 matches on the Italian sides and 444 on the German sides, which therefore become the sizes of the human evaluation sets of Ro $\rightarrow$ De and Nl $\rightarrow$ It, respectively, for which we can re-use the available post-edits.

3.3.4.3 Training Setting

Each of the three system types, namely *bilingual, multilingual and zero-shot*, is trained using both recurrent and transformer architectures, with the proper training data pro-

	encoder-decoder type	embedding size	hidden units	encoder depth	decoder depth	batch size
Recurrent	LSTM	512	1024	4	4	128 seg
Transformer	Self-Attention	512	512	6	6	2048 tok

Table 3.12: Hyper-parameters used to train recurrent and transformer models.

vided in the IWSLT 2017 evaluation campaign. Meta training parameters were set in a preliminary stage with the aim of maximizing the quality of each approach. Recurrent NMT experiments are carried out using the open source OpenNMT-py [83], whereas the transformer models are trained using the Tensor2Tensor toolkit. Hence, we took the precaution of selecting the optimal training and inference parameters for both approaches and toolkits. For instance, for our low-resource setting characterized by a high data sparsity, the dropout [150] is set to 0.3 [55] in recurrent and to 0.2 in transformer models to prevent over-fitting. Similarly, Adam [82] optimizer with an initial learning rate of either 0.001 (RNN) or 0.2 (transformer) is used. In all experiments, the training is run up to convergence. Table 3.12 summarizes the list of hyper-parameters.

3.3.4.4 Models

We train five types of models using either the Recurrent or the Transformer approaches. All models are trained up to convergence, eventually the best performing checkpoint on the dev set is selected. Table 3.13 summarizes the systems tested in our experiments. As references, we consider four bilingual systems (in short NMT) trained on the following directions: Nl→De/It and Ro→De/It. The first term of comparison is a many-to-many multilingual system (in short M-NMT) trained in all directions in the set {En,De,Nl,It,Ro}. Then, we test zero-shot translation (ZST) between related languages, namely Nl→De and Ro→It, by training a multilingual NMT without any data for these language pairs. We also test zero-shot translation between unrelated languages (ZST_A), namely Ro→De and Nl→It, by excluding parallel data between these languages. Finally, for the same unrelated zero-shot directions we also train multi-lingual systems (ZST_B) that include data related to Romanian and Italian, namely En↔Fr/Es.

Model	#Dir.	System Description
NMT	1	*Four bilingual models for the Nl → De/It and Ro → De/It directions.*
M-NMT	20	*Multilingual, trained on all directions in the set {En,De,Nl,It,Ro}.*
ZST	16	*Zero-shot, trained as multilingual removing Nl ↔ De and It ↔ Ro data.*
ZST_A	12	*Zero-shot, trained as ZST removing De ↔ Ro and Nl ↔ It data.*
ZST_B	16	*Zero-shot, trained as ZST_A adding En ↔ Fr/Es data.*

Table 3.13: *The training setting of 4*bilingual, 1*multilingual, and 3*zero-shot systems.*

3.3.5 Evaluation Methods

Systems are compared in terms of BLEU and TER [148] scores, on the single references of the official IWSLT test sets. In addition, two TER-based scores are reported, namely the multiple-reference TER (mTER) and a lemma-based TER (lmTER), which are instead computed on the nine post-edits of the IWSLT 2017 human evaluation set. In mTER, TER is computed by counting, for each segment of the MT output, the minimum number of edits across all the references and dividing by the average length of references. lmTER is computed similarly to mTER but looking for matches at the lemma level instead of surface forms. Significance tests for all scores are reported using Multeval [35] tool.

Systems are also compared in terms of three well known and widely used error categories, that is lexical, morphological, and word order errors, exploiting TER and post-edits as follows. First, the MT outputs and the corresponding post-edits are lemmatized and POS-tagged; for that, we used ParZu [141] for German and TreeTagger [132] for Italian. Then, the lemmatized outputs are evaluated against the corresponding post-edits via a variant of the *tercom* implementation[5] of TER: in addition to computing TER, the tool provides complete information about matching lemmas, as well as shift (matches after displacements), insertion, deletion, and substitution operations. Since for each lemma the tool keeps track of the corresponding original word form and POS tag, we are able to measure the number of errors falling in the three error categories, following the scheme described in detail in [14].

[5]Available at wit3.fbk.eu/show.php?release=2016-02&page=subjeval

3.3.6 Translation Analysis

3.3.6.1 Related Languages

First, we compare the bilingual (NMT), multilingual (M-NMT), and zero-shot (ZST) systems on the two tasks Nl → De and Ro → It, implemented as either recurrent or transformer networks, in terms of automatic metrics. As stated above, BLEU and TER exploit the official external reference of the whole test sets, while mTER and lmTER utilize the multiple post-edits of the (smaller) IWSLT human evaluation test set. Scores are given in Table 3.14.

	System	Recurrent				Transformer			
		BLEU	TER	mTER	lmTER	BLEU	TER	mTER	lmTER
Nl→De	NMT	18.05	64.61	23.70	20.60	**18.37**	**63.74**	27.95	23.86
	M-NMT	17.79	66.18	21.75	18.28	$^{\uparrow}$**19.95**	**61.90**	23.62	20.05
	ZST	17.06	65.73	26.35	22.29	$^{\uparrow}$**19.13**	**62.69**	**25.19**	**21.53**
Ro→It	NMT	22.16	59.35	22.99	20.39	**22.48**	**57.34**	26.60	23.36
	M-NMT	21.69	59.50	21.12	18.46	$^{\uparrow}$**22.12**	**57.51**	25.05	21.57
	ZST	18.72	62.08	29.66	26.15	$^{\uparrow}$**21.29**	**59.08**	**26.93**	**23.33**

Table 3.14: *Automatic scores on tasks involving related languages. BLEU and TER are computed on test*2017*, while mTER and lmTER are reported for human evaluation sets. Best scores of the transformer model against the recurrent are highlighted in bold, arrow* $^{\uparrow}$ *indicates statistically significant differences (*$p < 0.05$*).*

Looking at the BLEU/TER scores, it is evident that transformer performs better in all the three model variants. In particular, for the multilingual and the zero-shot models, the gain is statistically significant. On the contrary, the mTER and lmTER scores are better for the recurrent architecture; in this case, the outcome is misleading since the nine post-edits include those generated by correcting the outputs of the three Recurrent systems. As such, the translations of the recurrent systems are rewarded over the translations produced by the Transformer systems, thus making the comparison not fair.

As far as the models are compared, the bilingual one is the best in three out of four cases, the exception being the transformer Nl → De. Nonetheless, it is worth to note the good performance of the M-NMT model in terms of mTER and lmTER. This result holds

	System	Recurrent				Transformer			
		BLEU	TER	mTER	lmTER	BLEU	TER	mTER	lmTER
Ro→De	NMT	13.99	72.70	61.82	54.61	↑**16.52**	**66.71**	**55.68**	**48.44**
	ZST_A	14.93	69.38	58.26	51.08	↑**16.46**	**66.88**	**54.72**	**48.25**
	ZST_B	14.75	69.29	58.26	51.37	↑**16.55**	**67.18**	**55.29**	**48.03**
Nl→It	NMT	18.88	63.79	58.79	52.16	↑**20.22**	**60.88**	**55.52**	**48.56**
	ZST_A	18.77	62.97	58.80	51.32	↑**19.80**	**60.24**	**54.06**	**47.16**
	ZST_B	18.87	62.40	57.34	50.17	↑**20.61**	**59.41**	**53.04**	**46.17**

Table 3.15: *Evaluation results for the unrelated language directions. BLEU and TER scores are computed with single references, while mTER and lmTER are computed with nine post-edits.*

true in both recurrent and transformer approaches, regardless of the BLEU score. We hypothesize that the main reason behind this is the higher number of linguistic phenomena observed in training, thanks to the use of data from multiple languages, which makes the multilingual models more *robust* than the bilingual models.

3.3.6.2 Unrelated Languages

In unrelated language directions, our experimental setting is aimed at evaluating the impact of source *language-relatedness* with the target. Particularly, we focus on the zero-shot setup given its intrinsic difficulty, by taking the bilingual systems as references. Table 3.15 provides BLEU and TER based scores for the Ro → De and Nl → It directions.

Concerning the ZST_A training condition, in one case (recurrent Ro → De) it outstandingly allows to outperform the pure bilingual system, while in the other cases there is no significant difference between ZST_A and NMT, proving once again that zero-shot translation built on the "language-flag" of M-NMT is really effective [77]: in fact, at most a slight performance degradation is recorded as the number of pairs used in training decreases [156]. Although gains are rather limited, adding training data involving Romance target languages (Fr and Es, ZST_B) close to It impacts as hoped: ZST_B scores are in general better than both NMT and ZST_A in Nl → It, while they do not degrade with respect to ZST_A in Ro → De.

Similarly to what is observed for related pairs (Table 3.14), the transformer architecture shows definitely higher quality than the recurrent one, confirming the capability of the approach to infer unseen directions.

The overall outcomes from Tables 3.14 and 3.15 are: *i*) multilingual systems have the potential to effectively model the translation either in zero-shot or non zero-shot conditions; *ii*) zero-shot translation is a viable option to enable translation without training samples; *iii*) the transformer is the best performing approach, particularly in the zero-shot directions. The next section is devoted to a fine-grained analysis of errors made by the various systems at hand, with the aim of assessing the outcomes based on automatic metrics.

3.3.7 Fine-grained Analysis

As introduced in the evaluation methods in Sec. 3.3.5, now we focus on lexical, morphological, and reordering error distributions to characterize the behavior of the three types of models and the two sequence-to-sequence learning approaches considered in this work.

As anticipated in the previous section, it is expected that scores computed with reference to post-edits penalize transformer over recurrent systems because the outputs of the latter were post-edited, but not those of the former. We try to mitigate this bias by relying on the availability of multiple post-edits which likely allows to better match the transformer runs than a single reference would do. For the fine-grained analysis, we use instead the expedient of computing error distributions that are normalized with respect to the error counts observed in a bilingual reference system. In the next two sections, the fine-grained analysis is reported for related and unrelated languages pairs, consecutively.

3.3.7.1 Related Languages

Table 3.16 provides the distribution over the error types by the bilingual (NMT), multilingual (M-NMT), and zero-shot (ZST) models, implemented either with recurrent or transformer architectures, for the Nl $\rightarrow$ De translation direction. We also report, for each error type and M-NMT and ZST system, the observed relative difference of errors with respect to the bilingual reference model.

Nl→De	Recurrent					Transformer				
	NMT	M-NMT	Δ	ZST	Δ	NMT	M-NMT	Δ	ZST	Δ
Lexical	77.29	69.65	-7.64	83.73	+6.44	76.47	64.83	-11.64	69.53	-6.94
Morph	15.41	16.51	+1.10	19.1	+3.69	15.70	13.96	-1.74	14.13	-1.57
Reordering	5.53	3.14	-2.39	5.41	-0.12	6.20	4.97	-1.23	5.41	-0.79
Mor & Reo.	1.76	1.02	-0.74	1.61	-0.15	1.63	1.36	-0.27	1.53	-0.10
Total	100	90.31	-9.69	109.84	+9.84	100	85.12	**-14.88**	90.6	**-9.40**

Table 3.16: *Distribution of lexical, morphological, and reordering error types from the two MT approaches for Nl → De direction. Reported values are normalized with respect to the total error count of the respective bilingual reference model (NMT).* Δ *are variations with respect to the bilingual reference models (NMT).*

Considering each error category, we observe the same general trend for all systems: the lexical errors represent by far the most frequent category (76-77%), followed by morphology (15-16%) and reordering (3-6%) errors; cases of words whose morphology and positioning are both wrong, represent about 1-2% of the total errors. Beyond the similar error distributions, it is worth to note the variation of errors made by M-NMT and ZST models with respect to those of the NMT model: for the Recurrent architecture, there is a decrease of 9.69 and an increase of 9.84 points, respectively. On the contrary, the transformer architecture yields improvements for both models: total errors reduce by 14.88 and 9.40 points, respectively. The result for the transformer ZST system is particularly valuable since the average error reduction comes from remarkable improvements across all error categories.

For the Ro → It direction, results are given in Table 3.17. Although to a different extent, we observe a picture similar to that of Nl → De discussed above: lexical errors is the type of error committed to a greater extent, multilingual models outperform their bilingual correspondents (more for the Recurrent than for the transformer models), and ZST is competitive with bilingual NMT only if the Transformer architecture is adopted.

Training under the zero-shot conditions ZST_A and ZST_B assume less training data available and permit to measure the impact of introducing additional parallel data from related languages. We considered training conditions ZST_A and ZST_B here to perform Ro→It zero shot translation and report the outcomes in Table 3.18.

Ro→It	Recurrent					Transformer				
	NMT	M-NMT	Δ	ZST	Δ	NMT	M-NMT	Δ	ZST	Δ
Lexical	80.63	73.81	-6.82	102.79	+22.16	81.97	76.01	-5.96	84.12	+2.15
Morph	12.33	12.86	+0.53	16.00	+3.67	11.49	11.79	+0.30	12.44	+0.95
Reordering	5.74	3.71	-2.03	6.09	+0.35	5.35	4.64	-0.71	4.81	-0.54
Mor.& Reo.	1.30	1.15	-0.15	2.18	+0.88	1.19	1.09	-0.10	1.09	-0.10
Total	100	91.54	**-8.46**	127.07	+27.07	100	93.52	-6.48	102.45	**+2.45**

Table 3.17: *Distribution of the error types in the Ro → It direction for the recurrent and transformer approaches. From the variation of errors that compare M-NMT and ZST models with the bilingual reference (NMT), a larger margin of error is observed in case of transformer ZST model.*

Ro→It	Recurrent					Transformer				
	NMT	ZST_A	Δ	ZST_B	Δ	NMT	ZST_A	Δ	ZST_B	Δ
Lexical	80.63	108.27	+27.64	100.31	+19.68	81.97	82.11	+0.14	76.76	-5.21
Morph	12.33	17.11	+4.78	17.23	+4.90	11.49	13.09	+1.60	11.59	+0.10
Reordering	5.74	6.20	+0.46	6.16	+0.42	5.35	5.18	-0.17	5.59	+0.24
Mor. & Reo.	1.30	2.22	+0.92	2.30	+1.00	1.19	1.16	-0.03	1.02	-0.17
Total	100	133.81	+33.81	126	+26.00	100	101.53	**+1.53**	94.96	**-5.04**

Table 3.18: *Error distribution of ZST_A and ZST_B models for the recurrent and transformer variants. Transformer achieves the highest error reduction, particularly in the ZST_B model setting.*

Results show error counts for each condition normalized with respect to the corresponding bilingual NMT. The most interesting aspect comes from the fact that global variations in the normalized error counts of the zero-shot translation can be associated with the relatedness and variety of languages in the training data. As recently reported [156], zero-shot performance of recurrent models in a low resource setting seems highly associated with the number of languages provided in the training data. This is also confirmed by comparing performance of Recurrent models across the ZST (Table 3.17), ZST_A and ZST_B conditions. In particular, variations from the bilingual reference model, show significant degradation when some language directions are removed (from +27.07 to +33.81) and a significant improvement when two related languages are added (from

Ro→De	Recurrent					Transformer				
	NMT	ZST_A	Δ	ZST_B	Δ	NMT	ZST_A	Δ	ZST_B	Δ
Lexical	79.18	74.42	-4.76	74.09	-5.09	79.21	79.11	-0.10	78.52	-0.69
Morph	9.91	10.35	+0.44	10.07	0.16	9.92	10.05	+0.13	10.87	+0.95
Reordering	7.33	6.16	-1.17	6.16	-1.17	7.19	6.88	-0.31	7.22	+0.03
Morph. & Reo.	3.58	3.47	-0.11	3.47	-0.11	3.68	3.52	-0.16	3.60	-0.08
Total	100	94.4	**-5.60**	93.79	**-6.21**	100	99.55	-0.45	100.21	+0.21

Table 3.19: *Error distribution of the bilingual (NMT), ZST_A and ZST_B model runs for the unrelated Ro → De direction. The transformer moder shows the smallest sensitivity to the change in the number of training language pairs.*

+33.81 to +26.00).

Remarkably, the transformer zero-shot model seems less sensitive to the removal or addition of languages: actually a slight improvement is observed after removing Nl → It and De → Ro (ZST_A), i.e., from +2.45 to +1.53 total error reduction, followed by a large improvement when En → Fr/Es (ZST_B) are added, i.e. from +1.53 to -5.04. Notice that the latter results outperform the bilingual model. Increases or drops of specific error types with respect to the bilingual reference model show sharper differences across the different conditions. Overall, across all experiments, we see slight changes in the distribution of errors types.

For instance, the best performing transformer model (ZST_B in Table 3.18) seems to gain over the reference bilingual systems only in terms of lexical errors (-5.21). The zero-shot Transformer model trained under the ZST condition (Table 3.17) although globally worse than the bilingual reference, seems instead slightly better than the reference concerning reordering error (-0.54), which account for 5.35% of the total number of errors.

3.3.7.2 Unrelated Languages

In our second scenario, we evaluate the relative changes in the error distribution for the unrelated language directions (Ro → De and Nl → It). This section complements the translation results reported in Table 3.15, analyzing the runs from the ZST_A and ZST_B models in a different manner.

Nl→It	Recurrent					Transformer				
	NMT	ZST_A	Δ	ZST_B	Δ	NMT	ZST_A	Δ	ZST_B	Δ
Lexical	81.08	80.7	-0.38	78.79	-2.29	81.15	79.36	-1.79	77.48	-3.67
Morph	8.47	9.03	+0.56	8.38	-0.09	9.01	9.2	+0.19	9.03	+0.02
Reordering	7.78	6.63	-1.15	6.32	-1.46	7.51	6.74	-0.77	6.51	-1.00
Morph & Reo.	2.67	2.54	-0.13	2.38	-0.29	2.34	2.41	+0.07	2.45	+0.11
Total	100	98.89	-1.11	95.86	-4.14	100	97.71	**-2.29**	95.47	**-4.53**

Table 3.20: *Error distribution of the bilingual (NMT), ZST_A and ZST_B model runs.* Δ *shows the relative change in the error distribution of the zero-shot models with respect to the bilingual reference models.*

In the Ro → De unrelated direction (Table 3.19), the recurrent model shows a reduction in the error rate of 5.60 points (ZST_A) and 6.21 points (ZST_B) with respect to the bilingual (NMT) while for the transformer no significant differences are observed. These results confirm what observed in the automatic evaluation on the reference translations (Table 3.15). The gain observed by the Recurrent model on the ZST_B condition is mainly on lexical (-4.76 points) and reordering errors (-1.17 points) is probably due to the poor performance of its bilingual counterpart.

As far as the the Nl → It unrelated direction (Table 3.20) is concerned, both recurrent and transformer ZST models show to reduce the error counts over the bilingual reference model. Actually, a similar trend occurs in Ro → De (Table 3.19), but with a relatively higher error reduction in case of the transformer model. In particular, the transformer model shows reductions of -2.29 points for ZST_A and -4.53 for ZST_B, whereas for the recurrent model the improvements are slightly lower, namely -1.11 (ZST_A) and -4.14 (ZST_B) points. Remarkably, both the recurrent and transformer models benefit from additional training data related to Italian (compare ZST_A and ZST_B).

In conclusion, we observe that error counts of the zero-shot models in unrelated directions can increase (Table 3.19) when compared to the bilingual model. In related language direction, the most interesting aspect is observed with the discount of error in the Nl → It direction (Table 3.20). In particular, the ZST_B zero-shot model showed $> 2.0\%$ error reduction over the ZST_A model. This gain is directly related to the newly introduced training data (i.e., En ↔ Fr/Es) in case of ZST_B.

3.4 Conclusion

Following our findings in Sec 3.1 and 3.2 on the performance of low-resource and zero-resource translation within a multilingual setting, we aimed at analyzing the translation quality of the different systems. Hence, in this section, we showed how bilingual, multilingual, and zero-shot models perform in terms of overall translation quality, as well as the errors types produced by each system. Our analysis compared Recurrent models with the recently introduced Transformer architecture. Furthermore, we explored the impact of grouping related languages for a zero-shot translation task. In order to make the overall evaluation more sound, BLEU and TER scores were complemented with mTER and lmTER, leveraging multiple professional post-edits. Our investigation on the translation quality and the results of the fine-grained analysis shows that:

- Multilingual models consistently outperform bilingual models with respect to all considered error types, i.e., lexical, morphological, and reordering.
- The Transformer approach delivers the best performing multilingual models, with a larger gain over corresponding bilingual models than observed with RNNs.
- Multilingual models between related languages achieve the best performance scores and relative gains over corresponding bilingual models.
- When comparing zero-shot and bilingual models, relatedness of the source and target languages does not play a crucial role.
- The Transformer model delivers the best quality in all considered zero-shot condition and translation directions.

Our fine-grained analysis looking at three types of errors (lexical, reordering, morphology) shows significant differences in the error distributions across the different translation directions, even when switching the source language with another source language of the same family. No particular differences in the error distributions were observed across neural MT architectures (Recurrent vs. Transformer), while some marked differences were observed when comparing bilingual, multilingual, and zero-shot systems.

Chapter 4

Zero-Shot Neural Machine Translation

In this chapter, we present a novel zero-shot self-learning approach using multilingual NMT. We demonstrate how zero-shot translation for language pairs without parallel training data can be attend by exploring a multilingual model or a pivoting-based approach. In the next section (4.1), we begin by introducing our approach to improve zero-shot translation. In (4.2), we cover previous work on multilingual model for zero-shot translation, pivoting-based approaches and semi-supervised learning for a scenario in which parallel data are not available for model training, which leads us to motivate the adoption of our self-learning approach. Then (4.3), we discuss our zero-shot NMT modeling approach, which leverages semi-supervised learning to progressively improve zero-shot translation directions via incremental learning.

The experimental settings (4.4), cover two zero-shot directions (Italian $\rightarrow$ Romanian and Romanian $\rightarrow$ Italian) for which we compare the proposed approach against supervised NMT, zero-shot multilingual NMT, and pivot-based translation. We analyze the results and sample translations to show the effect of our incremental learning approach. Our experimental results are organized into two parts: first, we apply our approach by leveraging N language directions, then, we probe a more difficult zero-shot translation model, by simply reducing the number of language directions in the multilingual model. Our final results (4.5), confirm that our zero-shot NMT model outperforms conventional pivoting methods, up to matching the performance of a fully-trained bilingual system.

We conclude the chapter by emphasizing the importance of zero-shot NMT modeling, as a promising direction to improve translation directions with no available parallel data.

4.1 Introduction

As we have motivated in Chapter 1, MT of low-resource languages represents a challenge for neural machine translation (NMT) [88]. However, recent efforts in multilingual NMT (M-NMT) [77, 64] have shown to improve translation performance in low-resource settings. Besides resulting in performance gains for low-resource languages, the benefit of M-NMT is the possibility to perform zero-shot translation i.e., across directions that were not observed at training time.

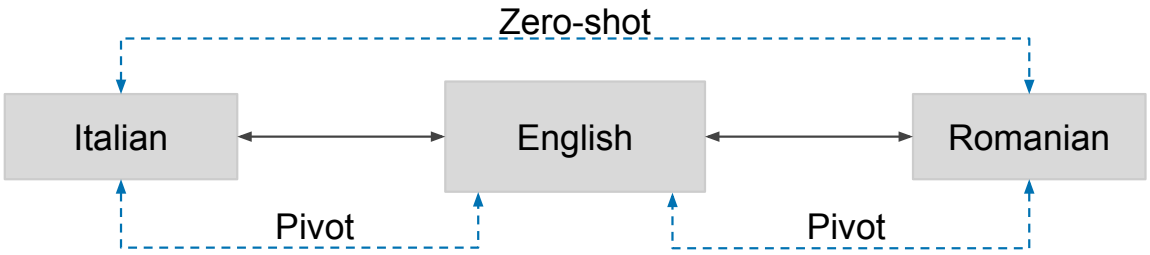

Figure 4.1: *Our zero-shot translation setting for Italian-Romanian. Parallel data is available only for Italian-English, Romanian-English, German-English, and Dutch-English. We leverage multilingual NMT trained on all available parallel data to translate across Italian ↔ Romanian, either directly (zero-shot) or through English (pivoting).*

Application scenarios in which zero-shot translation can bootstrap the creation of new parallel data – *e.g.* via human post-editing [77], show how translation performance in the initial zero-shot direction improves over time with the addition of new parallel data. We explore instead the possibility to enable a trained M-NMT model to further learn from its own generated data. Briefly, our method works as follows: first (1), we let the M-NMT engine generate zero-shot translations on some portion of the training data; then (2), we re-start the training process on both the generated translations and the original parallel data. We repeat this *training-inference* cycle until a convergence. Notice that, at each iteration, the original training data is augmented only with the last batch of generated translations (i.e., we discard the synthetic data utilized in the previous training stage). We observe that the generated outputs initially contain a mix of words from the shared

vocabulary, but after few iterations they tend to only contain words in the zero-shot target language thus becoming more and more suitable for learning.

We test our approach on an M-NMT scenario including Italian, Romanian, English, German and Dutch, assuming that the zero-shot translation pair is Italian-Romanian. We also make the assumption that all these languages have just parallel data with English (see Figure 4.1). We apply our approach on top of the multilingual NMT training method suggested by [77]. Experimental results show that our iterative training procedure not only significantly improves performance on the zero-shot directions, but it also boosts multilingual NMT in general. Moreover, our approach shows to outperform pivot-based machine translation, too.

4.2 Previous Work and Motivation

In this section, we summarize M-NMT, zero-shot translation, and discuss relevant works on model training with self-generated data.

4.2.1 Multilingual Model and Zero-Shot Translation

Previous work in M-NMT is characterized by the use of separate encoding and/or decoding networks for every translation direction. Dong et al. (2015) [43] proposed a multi-task learning approach for a *one-to-many* translation scenario, based on a sharing representations between related tasks – *i.e* source languages – in order to enhance generalization on the target languages. In particular, they used a single encoder on the source side, and separate attention mechanisms and decoders for each target language. In a related work, [102] used separate encoder and decoder networks for modeling language pairs in a *many-to-many* setting. Notably, they dropped the attention mechanism in favor of a shared vector space were to represent both text and multi-modal (such as speech) information.

In a *many-to-many* translation scenario, [52] introduced a way to share the attention mechanism across multiple languages. Despite the reported improvements, the need for an additional encoder and/or decoder for every language added to the system tells the limitation of these approaches, by making the resulting networks complex and expensive

to train. Most importantly, the multilingual approaches using language-specific encoder and decoder networks were not able to perform a direct zero-shot translation [53]. The main reason for this is related to the fact that, since each language is represented in a different semantic space, hence, it becomes difficult to trigger a target translation for a source without parallel training data. Though not being yet a zero-shot translation, the first zero-resource NMT was proposed by Firat et al. (2016) [53], by extending the M-NMT in [52]. The authors proposed a *many-to-one* translation setting and used the idea of generating a pseudo-parallel corpus [136], using a pivot language, to fine-tune the initial multilingual model (see Sec 2.3.7 for the details on zero-resource NMT modeling).

In a different way, [77] and [64] proposed an M-NMT approach by introducing a *target-forcing* mechanism – a *language-flag* that explicitly indicate the language of the output at time of training and inference (see Sec. 3.1 for details). Prepending language tokens has permitted to eliminate the need of separate encoder/decoder networks and attention mechanism for every language pair. Moreover, the single encoder-decoder model enabled a better zero-shot translation performance.

An attractive feature of the *target-forcing* mechanism is the possibility to perform zero-shot translation with the same multilingual setting as in [77, 64]. However, our findings in Chapter 3 indicate that the mechanism fails to achieve reasonable zero-shot translation performance. Particularly we identified two challenging scenarios: zero-shot languages that do not have large parallel data with a pivot language in the multilingual model (findings reported in Sec. 3.1), and varying degree of language relatedness in the multilingual model (as summarized in Sec. 3.2 and 3.3). The promising results in [77] and [64] hence require further investigation to verify if their method can work in various language settings, particularly across distant languages.

4.2.2 Semi-Supervised and Dual Learning

NMT model training using self-generated data has been around for a while. For instance, in statistical machine translation (SMT), [118, 11] showed how the output of a translation model can be used iteratively to improve results in a task like post-editing. Mechanisms like back-translating the target side of a single language pair have been suggested by Bertoldi et al. (2015) [15] for domain adaptation and more recently by Sennrich et al. [136]

to improve an NMT baseline model. These approaches based on back-translation can be collectively considered as *semi-supervised learning* methods as discussed in Sec. 2.3.7. In [180], a *dual-learning* (i.e., MT as a dual task, where the *source* $\rightarrow$ *target* is the primal task and the *target* $\rightarrow$ *source* is the dual task) mechanism is proposed where two NMT models working in the opposite directions provide each other feedback signals that permit them to learn from monolingual data. However, when it comes to language pairs with only monolingual data (zero-resource languages), both back-translation and dual-learning based semi-supervised approaches are not purposed to enable zero-shot translation, without an additional learning objective.

Hence, our approach aims to model zero-shot NMT from monolingual data in *dual zero-shot* translation (source $\leftrightarrow$ target) setting. Different from [140], and similar with [180] our approach leverages the capability of the network to jointly learn multiple translation directions. Although our brief survey shows that re-using the output of an MT system for further training and improvement has been successfully applied in different settings, our approach differs from past works in two main aspects: *i)* introducing for the first time a zero-shot NMT modeling mechanism (also know as *train-infer-train*) by utilizing an M-NMT model, and *ii)* casting the approach as a *self-correcting* training procedure over dual zero-shot directions, so that incrementally improved translations mutually reinforce each direction. In the next section, we go in detail describing our proposal on how to train a model for zero-resource languages with only monolingual data and incrementally improve the zero-shot translation directions.

4.3 Zero-Shot NMT Modeling

Our goal is to improve translation in the zero-shot directions of a baseline multilingual model trained on data covering N languages but not all their possible combinations (e.g. Italian $\leftrightarrow$ Romanian, Fig. 4.1). After training a baseline multilingual model with the target-forcing method [77], our self-learning approach works in the following way:

- First, a dual zero-shot inference (i.e., source $\leftrightarrow$ target directions) is performed utilizing monolingual data extracted from the training corpus.
- Second, the training is restarted by combining the zero-shot inference output and

the original multilingual data of the non-zero-shot language directions.

- Third, the *train-infer-train* cycle is repeated until a convergence point is reached on the dual zero-shot directions.

Algorithm 1: Train-Infer-Train	
INPUT: Data D, zero-shot languages (l_1, l_2), number of rounds N	
1: *M-NMT* $\leftarrow$ Train($\emptyset$, D)	pre-train multilingual model on D
2: $L_1 \leftarrow$ Extract(D, l_1)	extract l_1 monolingual data from D
3: $L_2 \leftarrow$ Extract(D, l_2)	extract l_2 monolingual data from D
4: **for** $i = 1$, N **do**	
5: $L_2^* \leftarrow$ Infer(*M-NMT*, L_1, l_2)	translate L_1 into l_2
6: $L_1^* \leftarrow$ Infer(*M-NMT*, L_2, l_1)	translate L_2 into l_1
7: $D^* \leftarrow D \cup (L_1^*, L_2) \cup (L_2^*, L_1)$	augment original data
8: *M-NMT** $\leftarrow$ Train(*M-NMT*, D^*)	re-train model on augmented data
9: **end for**	
10: return *M-NMT**	

Table 4.1: *Self-training algorithm for zero-shot directions* $l_1 \leftrightarrow l_2$.

4.3.1 Zero-Shot Self-Learning with Multilingual Model

Our training and inference strategy is summarized in Algorithm 4.1, while the flow chart in Fig. 4.2 further illustrates the training and inference pipeline. The proposed approach is performed in three steps, where the latter two are iterated for a few rounds. In the first step (line 2), a multilingual NMT system M is trained from scratch on the available data D ("*Train*" step). In the second step (lines 7-8), the last trained model M is run to translate ("*Infer*" step) between the zero-shot directions monolingual data L_1 and L_2 extracted from D (lines 3-4). Then, in the third step (line 10), training of M is re-started on the original data D plus the generated synthetic translations L_2^* and L_1^*, by keeping the extracted monolingual data L_1 and L_2 always on the target side ("*Train*" step). The updated model is then again used to generate synthetic translations, on which to re-train M, and so on.

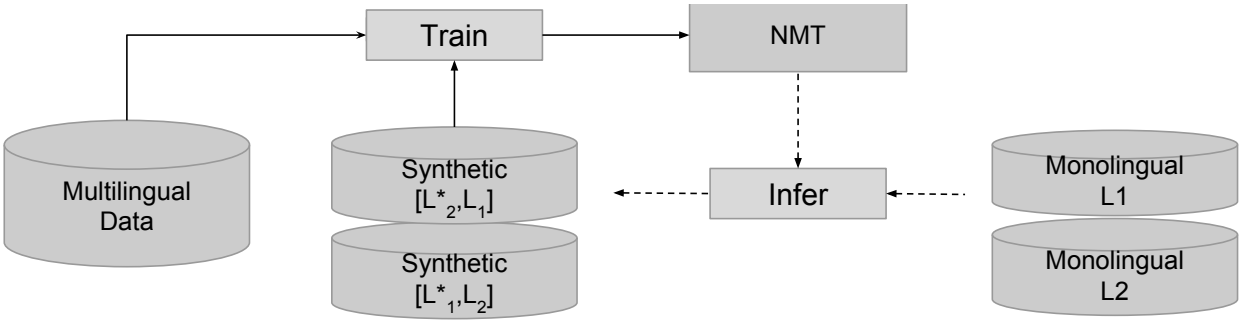

Figure 4.2: *Illustration of the proposed multilingual* train-infer-train *strategy. Using a standard NMT architecture, a portion of two zero-shot directions monolingual dataset is extracted for inference to construct a dual source↔target mixed-input and continue the training. The solid lines show the training process, whereas the dashed lines indicate the inference stage.*

4.3.2 Iterative Back-Translation and Mixed-Language Inputs

In the multilingual NMT scenario, the automatic translations used as the source part of the extended training data will likely contain a *mixed-language* that includes words from a vocabulary shared with other languages. This is likely due to the weak target-forcing signal at time of generation, which in turn depends on the performance of the M-NMT model for the initial zero-shot translation stage. The expectation is that round after round, the model will generate better outputs by learning at the same time to translate and "correct" its own translations by removing spurious elements from other languages. If this intuition holds, the iterative improvement will yield increasingly better results in translating between the source ↔ target zero-shot directions.

4.4 Experiments

4.4.1 Languages, Data, and Preprocessing

To evaluate our approach, we consider five languages: English (EN), Dutch (NL), German (DE), Italian (IT), and Romanian (RO). To simulate a low-resource scenario, each language pair has $\approx 200k$ parallel sentences (see Table 4.2 for details). Notice that the choice of this data is aimed at creating a low-resource NMT training scenario. All the

parallel datasets are from the IWSLT17 multilingual shared task [26].[1]

Direction	Train	Test 2010	Test 2017
EN ↔ DE	197,489	1,497	1,138
EN ↔ IT	221,688	1,501	1,147
EN ↔ NL	231,669	1,726	1,181
EN ↔ RO	211,508	1,633	1,129
IT ↔ RO	209,668	1,605	1,127

Table 4.2: *Number of sentences used to train the multilingual model on eight directions. The IT ↔ RO pairs are used to train only the bilingual models.*

To train all models, we used the same pipeline, After tokenization, we apply byte pair encoding (BPE) [137], using a jointly trained (on source and target data) shared BPE model to segment the tokens into sub-word units. For this operation we use $8,000$ BPE merging rules, with a minimum frequency threshold of 30 to apply the segmentation. When training the multilingual models, the pipeline includes adding the artificial language token at the source side of each parallel data to indicate the desired target language direction, both for the training and validation sets [77]. We evaluate our models using test2010, and for comparison we use test2017 from the IWSLT2017 evaluation dataset.

4.4.2 Training Details

All the experiments are carried out using the open source OpenNMT-py [83]. For training the models, we used the parameters specified in Table 4.3. Considering the high data sparsity of our low-resource setting, we applied a dropout of 0.3 [55] to prevent overfitting [150]. To train the baseline M-NMT, we used Adam [82] as the optimization algorithm with an initial learning rate of 0.001. In the subsequent *train-infer-train* rounds, we used SGD [39], with a learning rate of 1. If the perplexity does not decrease on the validation set or the number of epoch is above 7, a learning rate decay of 0.7 is applied. This combination of optimizers was found to be effective in accelerating the training in the first few iterations. In all the reported experiments the baseline models are trained until

[1]IWSLT2017 shared task: https://sites.google.com/site/iwsltevaluation2017/

Model parameters	Value
RNN type	LSTM
RNN size	1024
Embedding dim	512
Encoder	bidirectional
Encoder depth	2
Decoder depth	2

Table 4.3: *Hyper-parameters used to train all the models unless specified differently.*

convergence, while each train round after the inference stage is assumed to iterate over 10 epochs. For decoding, a beam search of size 10 is applied.

4.4.3 Models and Baselines

Our baseline models are trained both in a multilingual and bilingual settings. For each direction of the multilingual model and for every bilingual model we report the BLEU score computed using *multi-bleu.perl*. BLEU scores of the M-NMT systems trained on the parallel data in Table 4.2 are reported in Table 4.6 and 4.6 (second column). To compare our zero-shot translations against those of the bilingual models we trained two models: Italian→Romanian and Romanian → Italian. Both bilingual models are trained with the same amount of training data used by each direction of the M-NMT model (see Table 4.2). Moreover, as additional terms of comparison, we trained two pivoting-based systems (using *English* as a pivot language since it shares training data with Italian and Romanian): Italian → English → Romanian and Romanian → English → Italian.

4.4.3.1 Bilingual models

The baseline models for comparison consist of: an eight direction multilingual model (M-NMT), and two bilingual NMT models.

The results of the two bilingual models are shown in Table 4.4. From the M-NMT model (see row 9 and 10 of Table 4.6), we particularly focus on the performance in the zero-shot directions that can be compared with the results from these two models.

System	tst2010	tst2017
Italian→Romanian	19.66	19.14
Romanian→Italian	22.44	20.69

Table 4.4: *BLEU scores of two bilingual NMT models on IWSLT data tst2010 and tst2017*

Systems	tst2010	tst2017
Italian→ English →Romanian	16.4	15.00
Romanian→ English →Italian	18.9	17.36

Table 4.5: *Performance of the Italian ↔ Romanian pivot translation directions using English as a pivot on tst2010 and tst2017*

4.4.3.2 Pivoting Translation

If large scale parallel data are available between the zero-resource languages and the pivot language (e.g. Italian → English, Romanian → English), the pivoting strategy is the most intuitive way to accomplish zero-shot translation, or to translate from/into zero-resourced languages through high resource ones [177]. However, results in the pivoting framework are strictly bounded to the performance of the two combined translation engines, and especially to that of the weaker one (see Sec. 3.1 for details). In contrast, M-NMT models that leverage knowledge acquired from data for different language combinations (similar to multi-task learning) can potentially compete or even outperform the pivoting ones given a better language transfer within a single representation space. Checking this possibility is the motivation for our comparison between the two approaches.

In our experiment we take English as the bridge language between Italian and Romanian in both translation directions. Unsurprisingly, compared with those of the bilingual models trained on Italian ↔ Romanian data, the results shown in Table 4.5 are lower. On both translation directions, the bilingual models are indeed about 3.0 BLEU points better. Such comparison, however, is not the main point of our experiment. Our aim is in fact to fairly analyze performance differences between pivoting and zero-shot methods trained in the same condition in which Italian ↔ Romanian training data are not available.

4.5 Results and Analysis

Direction	**test2010**		**test2017**	
	M-NMT	M-NMT*	M-NMT	M-NMT*
English→Italian	27.07	**28.47**	29.02	**30.43**
Italian→English	32.12	**33.16**	32.87	**33.61**
English→Romanian	24.65	**25.37**	20.96	**21.94**
Romanian→English	32.7	**34.00**	27.48	**28.21**
English→German	26.39	**26.42**	19.75	**19.85**
German→English	31.3	**31.79**	24.12	**24.25**
English→Dutch	30.27	**30.85**	25.37	**26.12**
Dutch→English	35.13	**35.77**	**29.25**	29.15
Italian→Romanian	8.59	**17.38**	8.18	**17.08**
Romanian→Italian	8.65	**19.36**	8.58	**19.25**

Table 4.6: *Comparison on test2010 and test2017 set between a baseline M-NMT model against a M-NMT* model with our proposed train-infer-train approach for the Italian ↔ Romanian zero-shot direction. The best result for each direction is shown in bold.*

Table 4.6, shows the improvement on the Italian ↔ Romanian zero-shot directions using our approach (M-NMT*) against the baseline (M-NMT) model. Specifically, the Italian → Romanian direction reached 17.38 BLEU score improving over the baseline (8.59) by 8.79 points. Romanian → Italian translation improved with an even larger margin (+10.71) from 8.65 to 19.36 BLEU score.

In addition, and to our surprise, the results from our self-correcting mechanism showed to perform even better than the pivoting strategy. To check the validity of our results, we also compared the baseline multilingual system and our approach on the IWSLT 2017 test set (*test2017*). As shown in Table 4.6, the results confirm those computed on *test2010*, with almost identical gains (+8.9 and +10.67).

Besides the improvement on the zero-shot translation, the other important advantage of our approach is evidenced by the performance gains obtained on the language directions supported by parallel training corpora. In general, all translation directions have shown

improvements, except for the slight drop (-0.10 BLEU) observed for the Dutch → English direction in *test2017* case.

4.5.1 Iterative Training and Inference

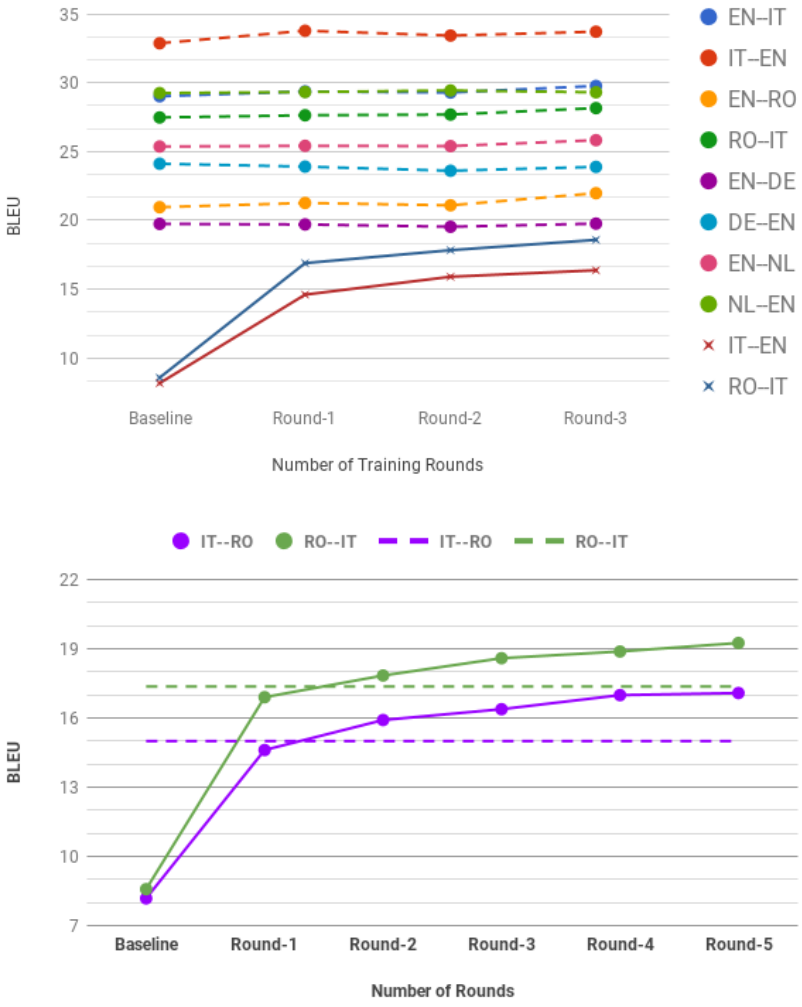

Figure 4.3: *Results from test2017 for the* 8 *non-zero-shot and the Italian ↔ Romanian zero-shot directions for three rounds (left), and comparing our iterative learning approach (solid lines) with the pivoting mechanism (dashed lines) for five rounds (right).*

Comparing the results at each round (see Figure 4.3), we observe that the training after the first inference step is responsible for the largest portion of the overall gain. This is mainly due to the initial introduction of (noisy) parallel data for the zero-shot directions.

The contribution of the self-correcting process can be seen in the following rounds, i.e., the improvement after each inference suggests that the generated data are getting cleaner and cleaner. As we have hypothesized in Sec. 4.3, the gain can be motivated by the fact that the feedback signal from the source → target translation direction helps to improve the dual target → source translation, and vice versa.

4.5.2 Outperforming Supervised NMT

In our second experimental setting, we investigate our zero-shot approach along two directions. First, we reduce the amount of language directions in the multilingual model creating a more difficult zero-shot translation task. Second, we replace the underlying model with the self-attention approach introduced in Sec. 2.3.3. Here, our final goal is to discover an efficient way of modeling zero-shot NMT, even for a worse-performing baseline multilingual model. We reduce the M-NMT language pairs from 8 to 4 directions, by dropping the Dutch ↔ English and German ↔ English parallel data. This implies that the new baseline multilingual model (M4-NMT) for the zero-shot task is trained only with the Italian ↔ English and Romanian ↔ English parallel segments. As additional baselines, we train single models for each language-specific direction (It → En, En → It, Ro → En, En → Ro, and It → Ro, Ro → It). Finally, by aggregating all the six language directions data, we also train a multilingual model (M6-NMT) as another comparison term.

The details of the evaluation setting and model parameters follow a similar strategy as in the first experimental setting, while the configuration for the Transformer model follows the settings in Sec. 3.3.

4.5.2.1 Bilingual, Multilingual, and Pivoting Results

Our first evaluation results are reported using the supervised bilingual (NMT) and multilingual (M6-NMT) models trained with the parallel segments of the Romanian ↔ Italian test directions. Then, we evaluate pivoting translation for the zero-shot directions using two bilingual models and the M4-NMT model, following pivoting through multilingual model as in Sec. 3.1 (see Table 3.3).

Direction	*Supervised*		*Pivoting*	
	NMT	M6-NMT	NMT	M4-NMT
Italian → Romanian	20.10	20.13	16.59	16.77
Romanian → Italian	21.36	21.81	17.87	19.39

Table 4.7: *Comparison of a supervised bilingual (NMT), multilingual (M6-NMT), and pivoting translations with bilingual models (NMT) and multilingual model (M4-NMT) on test2017.*

According to the results in Table 4.7, we observe a comparable performance between the supervised NMT and M6-NMT models. Moreover, although partial, the potential, of using multilingual models for pivoting unseen translation directions is visible from the M4-NMT model performance. The comparable results achieved in both directions speak in favor of training and deploying one system instead of two distinct NMT systems. Note that the performance of the final translation (i.e., pivot → target) is subject to the noise that has been propagated from the source → pivot translation step. This indicates that pivoting can be a favorable strategy if we have strong models from the source to the pivot and to the target.

4.5.2.2 Zero-Shot Translation

In this experiment, we show how our approach helps to significantly boost the baseline multilingual NMT model (M4-NMT). We run the *train-infer-train* for five consecutive rounds, each consisting of $2-3$ epochs of additional training on the augmented training data. Table 4.8 shows the improvements on the dual Italian↔Romanian zero-shot directions.

Direction	Zero-Shot	*Zero-Shot Ours*				
	M4-NMT	*R1*	*R2*	*R3*	*R4*	*R5*
Italian → Romanian	4.72	15.22	18.46	19.31	19.59	**20.11**
Romanian → Italian	5.09	16.31	20.31	21.44	21.63	**22.41**

Table 4.8: *Comparison between a baseline multilingual (M4-NMT) against the results from our proposed train-infer-train approach upto five rounds (R5).*

Direction	*NMT*	*M6-NMT*	$\Delta^{\text{M6-NMT}}_{\text{NMT}}$	*M4-NMT*	$\Delta^{\text{M4-NMT}}_{\text{NMT}}$	*R5*	$\Delta^{\text{R5}}_{\text{NMT}}$
Italian→Romanian	20.10	**20.13**	+0.03	4.72	-15.38	20.11	+0.01
Romanian→Italian	21.36	21.81	+0.04	5.09	-16.27	**22.41***	+1.05

Table 4.9: *Results summary comparing the performance of systems trained using parallel data (i.e., two single language pair NMT and a six direction multilingual M6-NMT systems) against the four directions multilingual baseline (M4-NMT) and our approach at the fifth round R5. Best scores are bold highlighted, whereas statistically significant ($p<0.05$) results in comparison with the baseline (NMT) are indicated with "$\star$".*

For both zero-shot directions, the larger gain using the M4-NMT model comes at *R1*. This is the first model trained after the inclusion of the dataset generated by the *dual-inference* stage. The It → Ro direction improves by +10.50 BLEU points from a 4.72 to 15.22, whereas Ro → It improves from a baseline score of 5.09 to 16.31 BLEU (+11.22). The contribution of the self-correcting process can be seen in the subsequent rounds, i.e., the improvements after each inference stage suggest that the generated data are getting cleaner and cleaner. With respect to the Transformer model pivoting results shown in Table 4.7, our approach outperformed both single pair and multilingual pivoting methods in the second round (*R2*) (see the third column of Table 4.8). Compared with the best performing multilingual pivoting, our approach at the fifth round (*R5*) has a +3.34 and +3.02 BLEU gain for the It → Ro and Ro → It directions respectively.

In addition to the visible gain with respect to pivoting mechanism, an interesting trend emerges when we compare our approach with the results of the language-specific single models and multilingual ones reported in Table 4.7. The summary in Table 4.9 shows the effectiveness of a dual inference mechanism in allowing the model to learn from its outputs. Compared to the models trained using parallel data (i.e., *NMT* and *M6-NMT*), our approach (*R5*) is either comparable (+0.01 BLEU in It → Ro) or better performing (+1.05 BLEU in Ro → It). The trend from the *train-infer-train* stages proves our hypothesis of modeling zero-shot NMT with the incremental learning procedure discussed in Sec 4.3.

Overall, our iterative self-learning approach showed to deliver better results than the bilingual counterparts within five rounds, where each round iterates for a maximum of three epochs. Indeed, the improvement from our approach is a concrete example to train

	Italian → Romanian
Source	... che rafforza la corruzione, l'evasione fiscale, la povertà, l'instabilità.
Pivot	... poarta de bază, evazia fiscală, sărăcia, instabilitatea.
M-NMT	... restrânge corupția, **fiscale de evasion, poverty, instabilitate**.
M-NMT*	... care rafinează corupția, **evasarea fiscală, sărăcia, instabilitatea**.
Reference	... care protejează corupţia, evaziunea fiscală, sărăcia şi instabilitatea.
	Romanian → Italian
Source	E o poveste incredibilă.
Pivot	È una storia incredibile
M-NMT	È una storia **incredible**.
M-NMT*	È una storia **incredibile**
Reference	È una storia incredibile .
	English → Italian
Source	We can't use them to make simple images of things out in the Universe.
M-NMT	Non possiamo usarli **per creare immagini semplici di** cose nell'universo.
M-NMT*	Non possiamo usarle **per fare semplici immagini di** cose nell'universo.
Reference	Non possiamo usarle per fare semplici immagini di cose nell'univero

Table 4.10: *Examples of zero-shot translations generated via English (Pivot), multilingual translation (M-NMT) and multilingual translation enhanced with the proposed zero-shot modeling (M-NMT*).*

models in a self-learning way, potentially benefiting language directions with parallel data, if cast in a similar setting.

4.5.3 Example Translations

Looking at the sample translation outputs using the different approaches in Table 4.10, we observe that the baseline M-NMT system produces mixed language output (*e.g.* "poverty" in Italian→Romanian and "incredible" in Romanian → Italian). With our approach the M-NMT* system instead tends to produce more consistent target language ("poverty" becomes "sărăcia" in Italian → Romanian and "incredible" becomes "incredibile" in Romanian → Italian). Furthermore, even in the non-zero-shot directions there are case

where the enhanced M-NMT* system produces better translations (see the last reported example).

4.6 Conclusion

In this chapter, we introduced a method to improve zero-shot translation in multilingual NMT under scarce training conditions. The proposed self-correcting procedure, achieved significant improvements over a multilingual NMT baseline and outperformed a pivoting approach. In particular, for enabling and improving zero-shot translation, we showed *i)* how multilingual pivoting can be used for achieving comparable results to those of multiple bilingual models, and *ii)* that our proposed self-learning procedure boosts performance of multilingual zero-shot directions by even outperforming both pivoting and conventional bilingual models.

The achieved results show the potential of the zero-shot NMT under different MT scenarios. For instance, assuming a better transfer-learning, enabling zero-shot translation between highly related languages, language varieties and styles with only monolingual data can be a promising direction. Note that in the current setting performance optimization across the various training and inference was not explored. Hence, subsequent works should focus on the efficiency of the approach to reach good performance in less time. Moreover, the current setup did not consider any form of selection for the dataset to be translated at the inference stage of the train-infer-train procedure. We expect that applying frequency and similarity-based approaches to select promising training candidates will bring further improvements.

Chapter 5

Transfer Learning for Low-Resource NMT

In the previous chapter we saw that the zero-shot translation of *zero-resource* languages can be improved using our incremental learning approach (*train-infer-train*). A problem that remains open is improving the translation task of languages with small parallel data (*low-resource*). Hence, in this chapter, we present our transfer-learning method across NMT models employing a dynamic vocabulary. Our approach allows extending a pre-trained model to new languages by adapting its vocabulary as long as new data becomes available (i.e., introducing new vocabulary items that are not included in the pre-trained model).

Based on the languages considered in the *parent* and *child* models, our approach is examined in two settings. In the first setting the parent model is pre-trained using a high-resource language pair (HRL). The parameter transfer mechanism is evaluated in two scenarios: *i)* simply adapt the parent model to the new language pair and *ii)* continuously add new language data to progressively grow to an M-NMT system. Our experiments spanning five low-resource languages (LRL) with constrained data sizes (5k and 50k parallel segments) for the child model show a significant performance gain, ranging from +3.85 up to +13.63 BLEU scores. In the second setting, the parent model is trained on multilingual data, and the adaptation to the child model is done either using data of a new language pair or by using more relevant data from other language pairs, based on a language relatedness data selection criterion. Our experiments show significant perfor-

mance gains with the dynamic adaptation of a pre-trained M-NMT system over baseline models; up to +17.0 BLEU with *relevant* data selection to the LRL, and +13.0 BLEU even with *zero* LRL data. We show how our method outperforms current approaches, such as massively multilingual models and data-augmentation on four LRL pairs.

In line with our motivation for the proposed approach, in the following sections, we begin by discussing related work on transfer-learning, pre-trained model adaptation, data-selection, and data-augmentation. Then, we discuss our dynamic transfer learning approach for LRL in two sections, respectively describing the method and discussing our experimental results.

5.1 Related Work and Motivation

5.1.1 Transfer Learning

Recent efforts [71, 127] in NLP research have shown promising results by applying transfer-learning techniques. Zoph and Knight [188] proposed transfer-learning technique across two NMT models to cope with the scarcity of training data. First, a "parent" model is trained with HRL pair data. Then, the encoder-decoder components are transferred to initialize the parameters of a LRL "child" model. In [114], the approach has been shown to perform better if languages between parent and child models are related. Both works, assumes a new LRL only on the source side, hence, at time of adaptation the decoder parameters are fixed. Later, in [84] the parent to child transfer-learning strategy has been shown to improve even for new languages on the target side.

Although our approach shares a common transfer-learning principle with [188], we diverge by hypothesizing to update all parameters as a beneficial strategy. Our transfer-learning approach relies on a dynamic vocabulary that enforces changes in the trainable parameters of the network in contrast to fixing them as in [188, 114]. In other words, our strategy is based on both the *source* $\rightarrow$ *target* and *target* $\rightarrow$ *source* translation directions that we consider as transferable. Moreover, we analyze our transfer-learning effectiveness based on a language relatedness assumption, between the parent and child model.

5.1.2 Adapting Pre-Trained Multilingual Models

Adapting a M-NMT model to a single language pair was first shown in [53], for a zero-resource NMT task. Later, a single attentional encoder-decoder multilingual model showed to be an efficient approach for low-resource and zero-resource settings [77, 64]. Transfer learning for MT task can be defined in two main forms; *i*) "vertically", aggregating data from several language pairs to train a single model (i.e. multilingual NMT) [77, 64], *ii*) "horizontally", pre-training a model with the available pairs and fine-tuning it using the test LRL (i.e. parent model → child model) [53, 187, 114]. Recently, the combination of the *vertical* and *horizontal* transfer approaches have shown promising results, for LRL translation task. Neubig et al. (2018) [113], demonstrated the effectiveness of the training a universal model (i.e., covering up to 58 LRL → English language directions), and adapting it with a LRL → English direction. The adaptation strategy shows a large performance gain, even in cases where the LRL → English is never seen in the pre-trained model.

Following, comparative results are reported on the same LRL → English test pairs. First, by improving the source side language representation and parameter sharing, [173] showed better adaptation performance than [113]. Alternatively, [1] outperformed the adaptation strategies in [113, 173] by training a many-to-many massive multilingual model. This tells us that different types of transfer-learning strategies still deserve further investigation.

5.1.3 Data Selection and Augmentation

In Chapter 4, we have discussed the significance of our dual back-translation based data-augmentation mechanism, to improve MT of low-resource and zero-resource languages. In [179], a general data-augmentation strategy is proposed to improve LRL pairs when adapting from a pre-trained M-NMT model. The approach leverages a target side monolingual and closely related HRL→English parallel data. Then, back-translation is used to generate a pseudoHRL from the monolingual data, while the HRL side of the parallel data is converted to pseudoLRL using word substitution from a bilingual dictionary [94], similar to the approach in [80]. The synthetic data is used to construct a pseudoHRL→English and a pseudoLRL→English pair. Finally, the synthetic data to-

gether with the small LRL→English data is used to improve over the baseline models. The approach, which creates additional parallel data for the adaptation stage outperformed the approaches reported for pre-trained model adaptation using test language and related language data [113, 173].

Overall, our approach shares a common ground on the effectiveness of pre-training a universal model and adapting it to ultimately improve LRL pairs, however, it differs on the following aspects: *i*) it only considers a scenario where all of the pre-trained models have never seen the test language pair, *ii*) it learns a language model on the LRL to select data from related languages, *iii*) it investigates the less explored direction of English→LRL translation, *iv*) it explores zero-shot translation without adapting the pre-trained model, and *v*) it employs a dynamic adaptation strategy.

5.2 Dynamic Transfer-Learning for Low-Resource Languages

In relation to how and when model vocabularies are built, there can be two distinct scenarios: In the first one, all the training data for all the language pairs are available since the beginning. In this case, either separate or joint sub-word segmentation models can be applied on the training material to build vocabularies that represent all the data [137, 178]. In the second scenario, training data covering different language directions are not available at the same time (most real-world MT training scenarios fall in this category), in which new data or new needs in terms of domains or language coverage emerge over time. In such cases, either: *i*) new MT models are trained from scratch with new vocabularies built from the incoming training data, or *ii*) the sub-word segmentation rules of a prior (parent) model are applied on the new data to continue the training as a fine-tuning task.

In all the scenarios, optimal sub-word segmentation is crucial to avoid out-of-vocabulary (OOV) tokens. However, different strategies for the different training conditions can result in performance degradations or longer training time [40]. More specifically, limiting the target task with the initial model vocabulary will result in: *i*) a sub-word segmentation that is unfavorable for the new language directions and *ii*) a fixed vocabulary/model dimension despite the varying language and training data size. We address these issues

by introducing a method to transfer knowledge across languages by means of a dynamic vocabulary. Starting from an initial model, our method allows to build new NMT models, either in a single or multiple language translation directions, by dynamically updating the initial vocabulary to new incoming data. For instance, given a pre-trained German-English system (L_1), the learned parameters can be transferred across models, while adopting new language vocabularies.

In our experimental setting we test two dynamic transfer-learning approaches:

- progAdapt: train a chain of consecutive NMT models by transferring the parameters of an initial model (L_1) to new language pairs (L_2 …L_N). In this scenario, the goal is to maximize performance for the new incoming pairs.

- progGrow: progressively introduce new language pairs to the initial model L_1 to create a growing M-NMT model that covers N translation directions. In this scenario, the goal is to maximize performance on all the language pairs.

In the next section, we introduce the notion of dynamic vocabulary for MT task across languages, then we layout the strategies for our *progAdapt* and *ProgGrow* based transfer-learning.

5.2.1 Dynamic Vocabulary

For the two transfer-learning scenarios, we update the vocabulary V_p of the pre-trained model with the new language pair vocabulary V_c. The approach simply keeps the intersection (same entries) between V_p and V_c, whereas replacing V_p entries with V_c if the entries of the former vocabulary do not exist in the latter. At training time, these new entries are randomly initialized, while the intersecting items maintain the embedding vectors of the former model. The alternative approach to dynamic vocabulary in a continuous model training is to use the initial model vocabulary V_p, which we refer to as static-vocabulary or static adaptation. For convenience, we refer to our approach as TL-DV - *Transfer-Learning using Dynamic Vocabulary.*

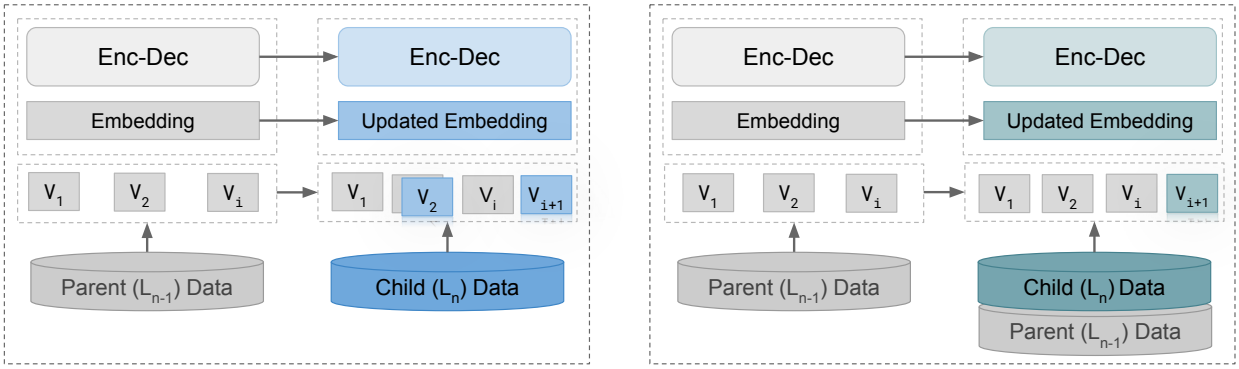

Figure 5.1: *Transfer-Learning, (left) progAdapt from a parent (L_{n-1}) to a child (L_n) model with new language pair, where parameters are transfered to the latter with the updated vocabulary and embedding space (i.e., keeping V_1 and V_i as overlapping entries, while replacing the non-overlapping V_2 of the parent and inserting new vocabulary V_{i+1} of the child.), and (right) progGrow from a parent model incorporating both the previous and the new language pair data and vocabulary entries.*

5.2.2 Progressive Adaptation to New Languages

In this scenario, starting from the pre-tarined model (L_1), we perform progressive adaptation by initializing the training of a model at each step (L_n) with the previous model (L_{n-1}). At time of reloading the model from L_{n-1}, a TL-DV update is performed as described above. In this approach, the data of the initial model is not included at the adaptation stage. We refer to the application of this approach in the experimental settings and discussion as *progAdapt*.

5.2.3 Progressive Growth of Translation Directions

In this scenario, a model at L_1 is simultaneously adapted to an incremental number of translation directions, under the constraint that the level of performance on L_1 has to be maintained. For a simplified experimental setup, we will incorporate a single language pair (source→target) at a time, when adapting to L_n from L_{n-1}. We refer to the application of this approach as *progGrow*.

5.2.4 Experiments

5.2.4.1 Dataset and Preprocessing

The experimental setting includes the pre-trained model language pair (German ↔ English) and three additional language pairs (Italian ↔ English, Romanian ↔ English, and Dutch ↔ English) for testing the proposed approaches. We use publicly available data from the WIT^3 TED corpus [25]. Table 5.1 shows the summary of the training, dev, and test sets. To simulate an extremely low-resource (M_{ELR}) and low-resource (M_{LR}) model settings, 5K and 50K sentences are sampled from the last three language pairs training data.

Language Direction	Train	Dev	Test	Data Received
German-English (De-En)	200k	1497	1138	L_1
Italian-English (It-En)	5k/50k	1501	1147	L_2
Romanian-English (Ro-En)	5k/50k	1633	1129	L_3
Dutch-English (Nl-En)	5k/50k	1726	1181	L_4

Table 5.1: *Data size of De-En pair for pre-trained model and the other pairs (It-En, Ro-En, Nl-En), assumed to be received progressively.*

At the preprocessing step, we first tokenize the raw data and in order to avoid training complexity we remove sentences longer than 70 tokens. As in [77], we prepend a "language-flag" on the source side of the corpus for all multilingual models. For instance, if a German source is paired with an English target, we append <2ENG> at the beginning of source segments. Next, a shared byte pair encoding (BPE) model [137] is trained using the union of the source and target sides of each language pair, with BPE segmentation rules of $8,500$ following [40]. At different levels of model (L_n) training, a new BPE model with respect to the specific language pairs is then used to segment the training, dev, and test data. Hence, depending on the lexical similarity and the overlap of sub-word units, the vocabulary size varies across consecutive model training steps, which in turn affects the embedding dimension.

5.2.4.2 Model and Settings

All systems are trained using the Transformer [168] model implementation.[1] We use LazyAdam, a variant of the Adam optimizer [82], with an initial learning rate constant of 2 and a dropout [150, 55] of 0.3. The learning rate is increased linearly up to 16,000 steps, and afterwards it is decreased with an inverse square root of the training step. To train our models we employ a network configuration of 512 hidden units and embedding dimension, and 6 layers of self-attention encoder-decoder network. The training batch size is of 4096 sub-word tokens. At inference time, we use a beam size of 5 and a batch size of 32. For a fair comparison, all baseline experiments are run for 100k steps, where all models are observed to converge. The consecutive experiments converge in variable training steps. However, to make sure a convergence point is reached, all restarted experiments on L_i are run for additional 50K steps. Systems are compared in terms of BLEU using the *multi-bleu.perl* implementation, on the single references of the official IWSLT test sets.

5.2.4.3 Baseline Models

The first baseline models (Bi-NMT) are trained in a bi-directional (i.e., source $\leftrightarrow$ target) setting for each of the language pairs in Table 5.1. In addition, we report scores from a multilingual (M-NMT) model trained with the concatenation of all the available language pair data, in other words we assume all the data is received at once.

5.2.5 Results

The performance of the proposed transfer-learning approach in the *progAdapt* and *progGrow* scenarios is reported for the low-resource (M_{LR}), and for an extremely low-resource data condition (M_{ELR}). In both dataset conditions, the proposed approaches are compared with the baseline systems (Bi-NMT and M-NMT In Table 5.2 and 5.3, the progAdapt results are reported from three consecutive adaptations to new language directions. These include the Bi-NMT De-En to It-En (L_2), followed by the adaptation to Ro-En (L_3), and then to Nl-En (L_4), whereas the progGrow is reported from the final adaptation stage (L_4). Bold highlighted BLEU scores show the best performing approach,

[1] OpenNMT: https://github.com/OpenNMT/OpenNMT-tf

	Direction	De-En	It-En	Ro-En	Nl-En
Bi-NMT (L_1)	>	**26.74**	25.21	10.80	21.75
	<	23.30	22.39	12.94	19.75
M-NMT (L_1)	>	24.14	26.42	22.17	24.00
	<	21.80	23.57	17.35	21.25
ProgAdapt ($L_{2/3/4}$)	>	-	$^{\uparrow}$**30.08**	$^{\uparrow}$**24.43**	$^{\uparrow}$**26.36**
	<	-	$^{\uparrow}$**26.24**	$^{\uparrow}$**20.31**	$^{\uparrow}$**25.52**
ProgGrow (L_4)	>	26.22	$^{\uparrow}$**29.61**	**23.23**	**24.78**

Table 5.2: *M_{LR} models performance i) at L_1 for the baseline (*Bi-NMT, M-NMT*) models, ii) at $L_{2/3/4}$ for* progAdapt*, and iii) at L_4 for the* progGrow *approach.*

while the $^{\uparrow\downarrow}$ arrows indicate statistically significant differences of the hypothesis against the better performing baseline (M-NMT) using bootstrap resampling ($p < 0.05$) [85]. The initial model (Bi-NMT) which is pre-trained with a data size 4× larger than M_{LR} and 40× the size of M_{ELR}, achieves BLEU scores of 26.74 and 23.30 respectively for the De → En and En → De directions.

5.2.5.1 Low-Resource Setting

The results of the baseline models shows that the M-NMT superiority over the Bi-NMT models in all directions except for De ↔ En. Compared to Bi-NMT and M-NMT performance all of the three progressive adaptations using the dynamic vocabulary technique (progAdapt) achieved a larger gain. If we look at the specific level of adaption (L_n) against the Bi-NMT, we observe that the It-En direction showed a +4.87 and +3.85 gain for the En and It target, respectively. When we take this model and continue the adaptation to Ro-En and Nl-En, we see a similar trend where the highest gain is observed on L_3 for the Ro-En direction with +13.63 and +7.37 points.

In case of the progGrow, where we focused our experiments on the English (En) directions only, we observed a similar improvement trend as in the progAdapt approach. The results are reported from the final stage (L_4) of the model growth, but improvements are consistent throughout the L_2 and L_3 stages. Overall, our observation is that the

proposed progGrow model can accommodate new translation directions when new data are received. Most importantly, improvements are observed for these newly introduced languages with only a slight drop (0.52 BLEU) in the De-En direction when compared with the initial model. Moreover, compared to the multilingual model as an alternative method for achieving cross-lingual transfer-learning, our approach shows improvements in the consecutive training stages.

5.2.5.2 Extremely Low-Resource Setting

	Direction	De-En	It-En	Ro-En	Nl-En
Bi-NMT (L_1)	>	**26.74**	7.64	4.56	5.69
	<	23.30	5.25	3.86	5.14
M-NMT (L_1)	>	24.96	**16.26**	**12.67**	**15.59**
	<	21.67	10.38	8.67	12.72
ProgAdapt ($L_{2/3/4}$)	>	-	$^{\downarrow}$15.16	$^{\downarrow}$11.03	$^{\downarrow}$11.52
	<	-	$^{\uparrow}$**14.40**	$^{\uparrow}$**11.10**	**13.57**
ProgGrow (L_4)	>	25.61	$^{\downarrow}$15.02	$^{\downarrow}$11.20	$^{\downarrow}$13.56

Table 5.3: *BLEU score for models using the M_{ELR} data.*

Similar to what we observed in the M_{LR} experiments, the baseline models in the M_{ELR} setting show poor performance. Looking at our approaches, we observe a higher gain at the first stage of progAdapt and progGrow. For instance, for the It-En pair there is a +7.52 improvement compared to the +4.87 in the M_{LR} models (see Table 5.2) over the Bi-NMT model. In the subsequent additional language directions (i.e., Ro-En and Nl-En), we also observe a similar trend. However, in comparison with the M-NMT, both of our approaches perform poorly when translating to the En target. The main reason for this could be the aggregation of all the available data for a single run in the M-NMT model, while our approaches exploit data when it becomes available in a continuous training. Alternatively the distance between each language pair could play a significant role when we adapt using an extremely sparse data.

5.2.6 Discussion

5.2.6.1 Effect of Language Relatedness

When related language pairs are consecutively added between two training stages (L_{n-1} and L_n), our TL-DV approach showed the best performance. For instance, for the Nl-En experiments, we changed the sequence of the added language pairs moving from a random order to a sequence based on the similarity to the initial pre-trained model (De-En).

		M_{LR}		M_{ELR}	
	Direction	De-En	Nl-En	De-En	Nl-En
ProgAdapt	>	-	$^{\uparrow}$**27.23**		**16.21**
	<	-	$^{\uparrow}$25.51		**15.86**
ProgGrow	>	26.62	$^{\uparrow}$**26.41**	26.52	$^{\uparrow}$**15.2**

Table 5.4: *M_{LR} and M_{ELR} models performance at L_2 for* progAdapt *and* progGrow *approaches in a closely related De-En (*Bi-NMT*) and Nl-En language pairs setting.*

The M_{LR} results confirm the general trend observed in Table 5.2, but, with a relatively better performance when translating in to English. Most importantly, the M_{ELR} results show a consistent and larger gain of +4.69 (Nl-En) and +2.29 (En-Nl) with the progAdapt, and +1.96 (Nl-En) with progGrow compared to the corresponding results in Table 5.3. Thus, the degree of language similarity is a direct influencing factor when incorporating a new language pair both in progAdapt and progGrow approaches.

Effect of Lexical Similarity: We further analyzed the influence of shared vocabularies between L_{n-1} and L_n models on the performance of TL-DV. For this discussion, we analyzed the progAdapt M_{LR} model from all adaptation stages. Fig. 5.2 (left) shows the improvement differences from consecutive models in relation to the % of shared vocabulary entries. For instance, L_1 (De-En) and L_2 (It-En) model vocabularies has a 47% overlap, whereas L_3 and L_4 share 53% and 51% with the previous model. The interesting aspect comes from the increase in model performance with a higher fraction of shared vocabulary entries. Thus, a larger number of shared parameters between two consecutive models allows for a larger gain in performance of the latter.

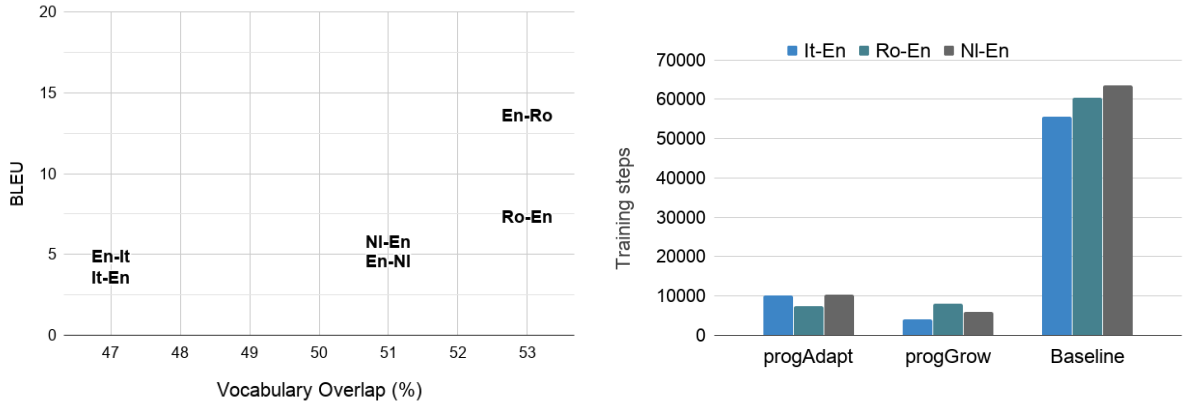

Figure 5.2: *The difference in performance between the baseline and* progAdapt *models in relation with the shared vocabulary between model L_{n-1} and new language pair model L_n (Left). Model training steps comparison for the three different language pairs between the baseline (rightmost) and the proposed approaches in the M_{ELR} setting (Right).*

5.2.6.2 Achieving Faster Convergence

The other main advantage of our TL-DV approach comes from the time a model takes to restart from the init model and reach a convergence point with better performance. In all experiments with our TL-DV approach, a converged model is found within 10K steps for M_{ELR} and 20K for M_{LR} training settings. Compared to ≈100K steps needed by a model trained from scratch to reach good performance, our approach takes only 4% to 20% of the training steps, with significantly higher performance. For instance, taking into consideration the M_{ELR} models, Fig. 5.2 (right) illustrates the steps required for the baseline systems to converge (Tab 5.3), in comparison with our approach where progGrow shows to converge slightly faster than progAdapt.

In summary, we emphasize that NMT models are not only data-demanding, but also require considerable time to be trained, optimized, and put into use. In particular real-world scenarios, strict time constraints prevent the possibility to deploy and use NMT technology (consider, for instance, emergency situations that require to promptly enable communication across languages [100]). On top of this, when the available training corpora are limited in size, delivering usable NMT systems (i.e., systems that can be used with the requirement of not making severe errors [14]) becomes prohibitive.

The results achieved by the transfer-learning with dynamic vocabulary approach in

two different training size conditions show that: *i)* adapting a trained NMT model to a new language pair improves performance on the target task, and *ii)* it is possible to train a model faster to achieve better performance. Overall, the capability of injecting new vocabularies for new language pairs in the initial model is a crucial aspect for efficient and fast adaptation steps.

5.3 Multilingual Model Adaptation to Unseen Languages

In the previous section, we explored the impact of tailoring a pre-trained model to a new language at adaptation time. The proposed dynamic transfer-learning showed to outperform baseline models with a large margin, especially when languages of the parent and child models are closely related. In this section, we further explore relevant data selection to be used in our transfer-learning approach. Moreover, different from the previous experimental scenarios, we now adapt a pre-trained multilingual (M-NMT) model to a specific unseen language pair(s). But first let us summarize our motivation, and discuss the evaluation settings of our hypothesis.

Among the reasons for the success of multilingual NMT is the positive cross-lingual transfer [153], which has been particularly beneficial for languages lacking large parallel data [77]. Although previous work document further improvements when using languages from the same family, they all rely on predefined linguistic assumptions about language similarity. Another challenge for facilitating access to information through M-NMT is that relevant LRL data might not be available at the time of training the initial seed model, or not available at all. In most real-life applications, new needs in terms of domains or language coverage arise continuously, making monolithic M-NMT models susceptible to out-of-vocabulary words. Moreover, new relevant training data in several (related or not) languages might become available continuously. Taking advantage of relevant data for adaptation is hence crucial to the performance of the final models [7, 175].

Recently, building a large scale M-NMT model was shown to be beneficial for LRL [1], even outperforming models specifically fine-tuned on the LRL data [113]. Another approach optimizes embeddings through character n-grams (i.e., soft decoupled encoding,

SDE) [173]. A more recent data augmentation approach showed improvements over all the previous approaches by adapting the M-NMT system using pseudo-bitexts generated by converting the HRL to the LRL [179]. Overall, research efforts in MT for LRL have shown that pre-training a multilingual NMT model and efficiently utilizing the available data are crucial towards better translation quality.

In this section, we motivate to choose segments from different related languages based on a perplexity measure. This is in a direct contrast to the approach in [113] that utilizes only the immediate related language, and data augmentation approach in [179], instead we aim to efficiently utilize multiple related language pairs by identifying the relevant examples to the test language pair.

More specifically, we investigate the usefulness of language similarity (distance between languages) as an indicator for selecting *which* and *how much* related HRL data can lead to the largest possible improvements. In analyzing these aspects, we examine the potential of a pre-trained universal (M-NMT) model in two settings: *i*) without having access to the test language data at training time (zero-shot translation), and *ii*) after adapting it to the LRL with selected data based on a language similarity criterion.

We evaluate our hypothesis in the following proposed settings:
Data Selection: We compute the perplexity of a LRL language model on available HRL data, in order to choose HRL data that are most similar to the LRL. Perplexity is a well-established information-theoretic measure, also used for measuring distance between languages [56]. We evaluate the data selection technique in different scenarios; including a) language family, b) random, and c) our proposed perplexity-based selection criterion.

Training and Inference: First we examine the performance of the universal multilingual model in total absence of LRL data (*zero-shot*). The evaluation involves both translation directions (LRL↔En). To date, model evaluation [113, 62, 173, 179] for the En → LRL has not been investigated yet. This direction is the most challenging one because of the small amount of available target side data in the LRL and the morphological richness of several LRLs compared to English.

Adaptation of Pre-Trained Model: We experiment with the adaptation of the multilingual NMT system by preserving the initial model vocabulary (*DirAdapt*) or dynamically updating it to include new items (*DynAdapt*), as introduced in Sec. 5.2. Fol-

lowing previous observations that more frequent segmentation favors morphologically rich languages and LRL [89, 28], we extend this approach by choosing different segmentation sizes that improve performance on the LRL.

Overall, our goal is to find a transfer-learning approach that leads to an efficient utilization of a pre-trained large-scale M-NMT model to improve the translation performance on the LRL pair. To achieve our goal of improving translation for the target LRL, we cast our approach as an unsupervised model adaptation strategy, in which relevant data for the adaptation are not supplied beforehand but have to be identified on the fly based on language similarity.

5.3.1 Data Selection by Language Distance

Our similarity criterion is based on perplexity, which is a commonly used measure to assess the quality of a language model [133], and has also been used to measure distance between languages [56]. We propose perplexity over popular data selection techniques in domain adaptation [7, 110], because the large number of languages involved makes training pairwise language models unfeasible. Thus, we use perplexity to select HRL data that are similar to the LRL data.

Perplexity is defined as the inverse probability of a test set (i.e., the HRL training data) computed using the LRL_{LM}. Thus, given the segments of the HRL set and the LM, the perplexity is computed as:

$$PP(S, LRL_{LM}) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1^{i-1})}} \tag{5.1}$$

S is a HRL segment consisting the sequence $w_1, w_2, \ldots, w_N$, P$(\cdot)$ are the n-gram probabilities estimated on the training set of LRL$_{LM}$. The distance between the LRL and the HRL is computed by evaluating the n-gram of the latter using the n-gram model of the former. For each HRL set, consisting of examples S_j, where $j = 1, \ldots, m$, we select S_j with the lowest perplexity (i.e., closest to the LRL) by computing $PP(S_j, LRL_{LM})$. We repeat the process for each HRL, re-score the sentences of all HRL based on their perplexity and select the necessary portion of data determined by a pre-configured threshold.

We train a language model on the LRL data (LRL_{LM}) and select training data with the lowest perplexity from related HRL (**Select-pplx**). We compare this approach with:

i. **Select-one** – select all available data only from one HRL related to the LRL as in [113].

ii. **Select-fam** – select all available data from a set of HRL related to the LRL belonging to the same language family.

iii. **Select-rand** – randomly sampling an equal proportion of data with Select-one and Select-pplx, from the HRLs that are closely related to the LRL.

5.3.2 Direct Vs. Dynamic Adaptation

For adaptation, we pre-process the test language data either *i*) using the pre-trained model's sub-word segmentation rules, or *ii*) by first learning a new sub-word segmentation model from the LRL data. Thus, for the transfer-learning stage, we follow two strategies:

1. **DirAdapt**: Vocabularies, sub-word segmentation rules and all parameters of the pre-trained model are used without any change.

2. **DynAdapt**: New vocabularies are generated using the new segmentation rule, and portions of the pre-trained model parameter are re-used.

In the *DirAdapt* case, the sub-word segmentation rules of the pre-trained model are applied on the test language for the inference and adaptation stages. The *DynAdapt*, instead, follows a similar dynamic adaptation strategy as the one discussed in Sec. 5.2, however, we first look for the test language segmentation that maximizes the overlap with the pre-trained model vocabularies.

Zero-shot Translation:
We specifically aim at assessing the potential of the large scale M-NMT model towards zero-shot translation (ZST). Unlike with adaptation strategies, the translation is evaluated in an extreme scenario, where the LRL has never been seen at training time. This means that the transfer-learning to assist the LRL translation is expected to come from multiple

languages, particularly related languages, that are present in the pre-trained model. We examine both a $\text{LRL}_{unseen} \leftrightarrow$ HRL translation directions, where:

1. $\text{LRL}_{unseen} \rightarrow$ HRL: represents a condition where at training time the source side only sees related languages to the LRL, and no LRL data at all.

2. HRL $\rightarrow \text{LRL}_{unseen}$: represents a so-far unexplored and more challenging condition, as motivated at the beginning of this section.

To evaluate the two scenarios we pre-train several (parent) models with data featuring different size and language combinations, to help us better understand the effect on the child models. For constructing the adaptation data, we follow the perplexity-based data-selection criterion. As such our objectives are; *i)* to evaluate how pre-trained models perform before an adaptation stage on unseen test language data, and *ii)* how models trained on data with different levels of language relatedness behave in addressing a zero-shot translation direction.

Our expectation is that the more closely related language pairs (HRL) to the test language (LRL_{unseen}) are available, the higher the performance of the pre-trained models will be. Comparing the zero-shot translation against the adapted models using a similar data selection criterion and data of the LRL_{unseen} will shade light on how much the pre-training helps. Moreover, the zero-shot translation can signal how robustly both the encoder and the decoder learn from different combinations of related languages, without seeing the test language.

5.3.3 Experiments

Data and Preprocessing:
We use four LRLs paired with English (En) for evaluating the two adaptation strategies: including Azerbaijani (Az), Belarusian (Be), Galician (Gl), and Slovak (Sk). Turkish (Tr), Russian (Ru), Portuguese (Pt), and Czech (Cs) as the closest related HRL respectively. All languages are first used to train the massive M-NMT model, except for the language serving as test language at each time. The choice of the test languages facilitates comparisons with previous works on similar settings. As a second step, we select a single

test language pair (En↔Gl) for an in-depth analysis of the data selection strategies, the zero-shot inference and the adaptation approaches. The data set size of the four LRL with linguistically closest HRL are used as in [113]. Data preprocessing follows a similar strategy as in Sec. 5.2.

Model and Settings:
The LM used for data selection is a 1-layer LSTM model with embedding and hidden layer of 512 units. We found that the best results were obtained when keeping all other settings as proposed in [170] (small model). The NMT models are implemented using the Transformer architecture [168], model training settings follows the experimental setup in Sec. 5.2.

5.3.3.1 Measuring Language Distance

For the related language data selection method, we focus on one language, Galician (Gl) as the test language, paired with Portuguese (Pt), in addition to Spanish (Es) and Italian (It) as further auxiliary languages. First, we select Pt as the closest language to Gl (*Select-one*). Then, we include Pt+Es and Pt+Es+It for the experiments with selection based on the language family (*Select-fam*).

Gl	Size	Lang	Select-fam	Select-pplx
Train	10k	Pt	184k	98.65k
Dev	682	Es	196k	79.51k
Test	1,007	It	204k	6.85k
		Total:	584k	184k

Table 5.5: *Data size in sentences for LRL (Gl), and selected data from HRL's (Pt, Es, It) with Select-fam and Select-pplx data selection strategies.*

For *Select-pplx*, we train a neural language model[2] on the Gl data to re-score sentences from the training corpora of related languages and select the sentences with the lowest perplexity until we match the corpus size of Pt. Selection is made without replacement,

[2]LM toolkit: https://github.com/lverwimp/tf-lm

i.e., an English sentence can have translations in multiple languages. Statistics are shown in Table 5.5. In order to check if the improvement observed by adding more languages is due to simply having more training data, we set a maximum limit at time of data selection. We hence select the same amount of data (i.e., 184k for the case of Gl) using each of the following approaches: Select-one, Select-rand, and Select-pplx.

5.3.3.2 Baselines and Comparison

Single language pair models are trained from scratch using only the test LRL data. First, results from the adaptation and data-selection strategies are compared with these baselines. Then, we compare against solutions previously proposed in literature (see Sec. 5.1 for details), namely:

- A static adaptation of the multilingual model to the LRL, *RapAdapt* [113] and *SDE* [173].
- A massive multilingual model trained including all the test LRL, avoiding adaptation (*Many* $\leftrightarrow$ *Many*) [1].
- A data-augmentation for LRL pair, followed by adaptation of a multilingual model (*Data-Augment*) [179].

In the first case (*RapAdapt, SDE*), a similar strategy to our *DirAdapt* is implemented using an RNN model. For a fair comparison with our Transformer-based approach, we take the relative improvement (Δ) between the single pair baselines and the dynamically adapted models. The second (Many$\leftrightarrow$Many) and third (*Data-Augment*) approaches utilize the Transformer model, allowing us to directly compare against the reported results. More interestingly, these comparisons bring together several approaches using the same four test languages, aiming at improving the quality of LRL translation. As a metric to evaluate translation quality, we use BLEU.

	Strategy	Az[Tr]	Be[Ru]	Gl[Pt]	Sk[Cs]	AVG.
Neubig & Hu 2018 Wang et al., 2018	Baseline	2.70	2.80	16.20	24.00	11.43
	M-NMT→Bi (RapAdapt)	10.70	17.40	28.40	28.00	21.20
	M-NMT→Bi (SDE)	11.82	18.71	30.30	28.77	22.40
	Δ (SDE-Baseline)	9.12	15.91	14.10	4.77	10.98
Aharoni et al., 2019	Many ↔ Many	12.78	21.73	30.65	29.54	23.67
Xia et al., 2019	Data-Augment	**15.74**	**24.51**	33.16	32.07	26.37
Ours	Baseline	3.61	4.42	16.32	26.44	12.70
	M-NMT→Bi (DirAdapt)	14.43	22.06	33.53	30.13	25.04
	M-NMT→Bi (DynAdapt)	15.33	23.80	**34.18**	**32.48**	**26.45**
	Δ (DynAdapt-Baseline)	**11.72**	**19.38**	**17.86**	**6.04**	**13.75**

Table 5.6: *BLEU scores for the four LRL → En comparing against previous approaches; RapAdapt [113], SDE [173], Many ↔ Many [1], and Data-Augment [179]. Bi is an adaptation with the LRL + [closest-HRL] according to Select-one strategy.*

5.3.4 Results and Analysis

5.3.4.1 Direct Vs. Dynamic Adaptation

In Table 5.6, we show the the relative improvement (Δ) between the baseline of [113] (*RapAdapt*) and the best performing adaptation approach (*SDE*), against the Δ between our baseline and our best performing approach (*DynAdapt*). Even with stronger baselines, our Δ is higher than in the previous approaches with +2.77 BLEU averaged over the four test languages. Note that the M-NMT model refers to a training setting with all except for the test language (cold start). We also note that our *DirAdapt* outperformed the *RapAdapt* and *SDE* with a larger margin in all the test languages.

Aharoni et al. [1] argue that the better performance of *Many↔Many* over the *RapAdapt* and *SDE* is due to avoiding model over-fitting by including more languages on both the encoder and decoder sides. However, our adaptation strategies show better performance in all test cases, with a +1.37 (*DirAdapt*) and +2.78 (*DynAdapt*) average BLEU. In fact, the additional improvement from *DirAdapt* comes from tailoring the segmentation for the

	Strategy	Az	Be	Gl	Sk
Neubig & Hu	Select-one	3.80	2.50	8.60	5.40
	M-NMT	3.70	3.50	15.50	7.30
Ours	Select-one	3.25	2.07	13.59	9.30
	M-NMT	**11.06**	**10.97**	**27.28**	**20.57**

Table 5.7: *Results for LRL → En ZST using model trained with a single pair Select-one, and all but the test LRL (M-NMT).*

test language and partially transferring the M-NMT model parameters.

By contrasting the performance of previous works against the *DynAdapt*, we learn that our method is superior to all, in average BLEU score. Specifically, when compared to the latest *Data-Augment* [179], the *DynAdapt* shows better performance in two of the test languages (Gl, Sk), and slight degradation for Az and Be. Our motivation for the lower performance is that the data augmentation results in much larger synthetic data, while our adaptation utilized only the original LRL data for each of the test languages and the closest related language pair (amounting to a max of $200k$ segments) as in [113]. Overall, our approach showed the possibility to achieve better performance when initializing from pre-trained M-NMT parameters.

5.3.4.2 Pre-Training for Zero-shot Translation

Comparing the approaches in [113] that used RNNs for evaluating the ZST settings against our results, we observe a large difference (see Table 5.7) that again attests the superiority of the Transformer model. The better performance is particularly true for the M-NMT models that are trained using all the available data but the test language. Previous works have also shown similar findings for the Transformer model when it comes to zero-shot translation [154, 1]. Thus, it is important to emphasize that the multilingual model is the best suit for further investigation by applying the data-selection procedures with the two adaptation options.

5.3.4.3 Data Selection for Zero-Shot Translation

Table 5.8 shows results for ZST using various data-selection strategies. In the Gl→En direction, adding more data from related languages improves performance but the improvement slows down as more languages are added. Even without any test language data, performance increases from 13.59 BLEU for training only with pt (*Select-one*) to 24 BLEU for Pt+Es (*Select-fam*), while with it (Pt+Es+It) increases further by 1.34 BLEU.

	Strategy	Gl→En	En→Gl
Our non ZST	Baseline	16.32	**11.83**
Ours ZST	Select-one	13.59	8.05
	Select-rand	14.69	4.09
	Select-pplx	15.55	5.38
	Pt+Es	24.17	4.61
	Pt+Es+It	25.51	4.17
	M-NMT	**27.28**	**8.78**

Table 5.8: *BLEU for ZST using models trained with different data-selection criteria. Pt+Es and Pt+Es+It are the two varieties of the Select-fam method.*

The M-NMT model scores higher, but only by 1.77 BLEU when compared to best *Select-fam* strategy. Here, it is important to emphasize that: *i*) the M-NMT model is trained using over 5M segments except for the test (Gl-En) pair, meaning that the performance of *Select-fam* shows the possibility to improve a ZST by having more related languages but less data, *ii*) while the amount of data is the same for *Select-one*, *Select-rand*, and *Select-pplx*, the latter shows better performance, indicating the importance of the data selection criteria using perplexity.

Opposite results are obtained in the En→Gl direction. As expected, our second evaluation of ZST into an unseen language on the decoder side does not perform well and performance decreases as more languages are added (i.e., from Pt+Es to Pt+Es+It). However, selecting related-language data using *Select-pplx*, we observe a better performance among the data-selection criteria at 5.38 BLEU.

Overall, ZST performance when translating from an unseen source language (Gl) into

a seen target language (En) is better than the baseline (see Table 5.8), with more than 10.0 BLEU points. This gain highlights the importance of closely related languages for improving the performance on the LRL. However, the opposite direction, where we infer into unseen target language (Gl), is a more challenging task that requires further investigation and the availability of at least monolingual data for the LRL.

5.3.4.4 Data Selection Strategies for Adaptation

M-NMT Adaptation	Gl→En	En→Gl
Strategy	Dir/Dyn- Adapt	Dir/Dyn- Adapt
→ *Gl*	32.18 / -2.5	26.39 / -3.21
→Select-one + *Gl*	33.53 / +0.65	26.45 / +0.28
→Select-rand + *Gl*	32.61 / +0.75	25.94 / +0.06
→Select-pplx + *Gl*	$^{\uparrow}$**34.15** / $^{\uparrow}$**+1.41**	$^{\uparrow}$**27.35** / $^{\uparrow}$**+0.59**
→pt+es+it + *Gl*	33.38 / +2.14	26.40 / +1.14

Table 5.9: *BLEU using models adapted from the M-NMT in different data conditions.* $^{\uparrow}$ *indicates statistically significance using bootstrap re-sampling* ($p < 0.05$) *[85].*

Table 5.9 shows results for adapting a M-NMT model with data selected using our proposed perplexity-based method, both in the direct and the dynamic adaptation scenario. As a general rule, adaptation with selecting data from several languages improves over adapting only with the target language. One possible reason for this improvement is avoiding over-fitting to the little data of the target language, as shown in [113]. However, perplexity-based data selection (*Select-pplx*) outperforms selecting only one related language (*Select-one*) for both translation directions. Moreover, we show that the improvement does not come only from mixing several related languages, since *Select-rand* hurts performance for both directions. Our method improves even over adapting with all data from the most related languages (*Pt+Es+It*), allowing for a faster adaptation.

The results show that perplexity can be a reliable measure for selecting smaller amounts of related-language data both in translation and adaptation from a M-NMT model in order to obtain larger improvements. For data selection strategies (either with

perplexity or random), better performance is achieved with faster convergence. This confirms that the data-selection and the adaptation strategy is the fastest way to build a usable and better performing system for an unseen language from a pre-trained model.

Comparing the results of *DirAdapt* and *DynAdapt*, the latter shows consistent improvements when adaptation is performed with data from at least two languages (LRL+another language). This can be attributed to the fact that the *DirAdapt* has a complete overlap (100%) both for the source and target side vocabularies with the pre-trained initial model (i.e., the initial model vocabulary is used without any modification), as well as the transfer of all parameters when adapting. On the contrary, the *DynAdapt* improvement comes from a careful sub-word segmentation of the test language before adaptation, resulting in a new vocabulary and consequently enforcing a partial transfer of parameters from the initial model. In addition to the importance of data-selection, the additional gain using *DynAdapt* indicates that a universal multilingual model can be made stronger if tailored to the characteristics of the test languages when adapting.

When conducting a qualitative evaluation of the sub-word segmentation, for extremely low-resource test languages (such as Gl↔En with $10k$ and Az↔En with $5k$ bitext), we observed a frequent segmentation that favours sub-words closer to character level for most of rare words included in the vocabulary. This is consistent with previous work supporting character-level segmentation for improving NMT of LRL [89, 28]. Furthermore, with a reduced vocabulary size, *DynAdapt* can compress the model with smaller embedding and pre-softmax linear transformation dimensions compared to the pre-trained model, and with sharing all the updated weight matrix as in [126].

5.4 Conclusion

In this chapter, we proposed a transfer-learning approach based on a dynamic vocabulary strategy when adapting from a parent model to a child model. We showed that, with our transfer learning approach, it is possible to train a faster converging model that even achieves better performance. Investigation into transfer-learning between related languages confirms previous work, and shows better performance for low-resource and zero-resource languages.

Moreover, we proposed and demonstrated adaptation from a pre-trained model using data selection strategies. To this aim, we used perplexity to select the most relevant data to the test language. We showed that perplexity-based data selection improves translation, leading to an improvement up to 10.0 BLEU points for LRL $\rightarrow$ En and 17.0 BLEU points for En $\rightarrow$ LRL when adapting from a universal multilingual model. Our adaptation strategy with selected data is useful even in the extreme case of zero-shot translation for an unseen language (+13.0 BLEU). Overall, we showed that our dynamic vocabulary based transfer-learning enforces changes in the trainable parameters of the network in contrast to fixing them as done with a static adaptation.

Chapter 6

NMT into Language Varieties

In this section, we investigate the problem of training NMT to translate into language varieties, assuming both labeled and unlabeled parallel texts. Both research and commercial machine translation have so far neglected the importance of properly handling the spelling, lexical and grammar divergences occurring among language varieties. Notable cases are standard national varieties such as Brazilian and European Portuguese, and Canadian and European French, which popular online machine translation services are not keeping distinct. We show that an evident side effect of modeling such varieties as unique class is the generation of inconsistent translations. We compare language variety-specific and generic NMT baselines against multilingual NMT systems, exploiting manual as well as automatic language variety labels. We show significant BLEU score improvements over baseline systems when translation into language varieties is learned as a multilingual task with shared representations.

We present systematic ways to approach NMT from English into four pairs of language varieties: Portuguese European (pt-EU) - Portuguese Brazilian (pt-BR), European French (fr-EU) - Canadian French (fr-CA), Serbian (sr) - Croatian (hr), and Indonesian (id) - Malay (ms)[1]. For each couple of varieties, we assume to have both parallel text labeled with the corresponding couple member, and parallel text without such information. Moreover, the considered target pairs, while all being mutually intelligible, present

[1]According to Wikipedia, Brazilian Portuguese is a dialect of European Portuguese, Canadian French is a dialect of European French, Serbian and Croatian are standardized registers of Serbo-Croatian, and Indonesian is a standardized register of Malay.

English (source)	I'm going to the gym before breakfast. No, I'm not going to the gym.
pt (GoogleTranslate)	Eu estou indo para a academia antes do café da manhã. Não, eu não vou ao ginásio.
pt-BR (M-C2)	Eu vou á academia antes do café da manhã. Não, eu não vou à academia.
pt-EU (M-C2)	Vou para o ginásio antes do pequeno-almoço. Não, não vou para o ginàsio.
pt-BR (M-C2_L)	Vou à academia antes do café da manhã. Não, não vou à academia.
pt-PT (M-C2_L)	Vou ao ginásio antes do pequeno-almoço. Não, não vou ao ginásio.

Table 6.1: *MT from English into Portuguese varieties. Example of mixed translations generated by Google Translate (as of 20th July, 2018) and translations generated by our variety-specific models. For the underlined English terms both their Brazilian and European translation variants are shown. Note, gym is academia in pt-BR and ginásio in pt-EU, whereas breakfast is café da manhã in pt-BR and pequeno-almoço in pt-EU.*

different levels of linguistic similarity and also different proportions of available training data. For our tasks we rely on the TED Talks collection[2], used for the International Workshop of Spoken Language Translation, and OpenSubtitles2018, a corpus of subtitles available from the OPUS collection[3].

In the sections to follow, we first provide a concrete motivation and define the challenges of translating into language varieties. Then, we review existing approaches to handle NLP and MT of dialects and related languages. Hence, we discuss baseline NMT systems, either language/dialect-specific or generic, and multilingual NMT systems, either trained with fully supervised (or labeled) data or with partially supervised data. We introduce our data sets, NMT set-ups based on the Transformer architecture, and then present the results for each evaluated system. We then conclude with a discussion and summary of our findings.

6.1 Motivation and Problem Statement

While just few years ago research in MT was struggling to achieve *useful* translations for the most requested and high-resourced languages, the level of translation quality reached today has raised the demand and interest for less-resourced languages and the solution of

[2]TED talks: http://wit3.fbk.eu/
[3]Opus: http://opus.nlpl.eu/

more subtle and interesting translation tasks [14]. If the goal of machine translation is to help worldwide communication, then the time has come to also cope with dialects or more generally language varieties[4]. Remarkably, up to now, even standard national language varieties, such as Brazilian and European Portuguese, or Canadian and European French, which are used by relatively large populations have been quite neglected both by research and industry. Prominent online commercial MT services, such as Google Translate and Bing, are currently not offering any variety of Portuguese and French. Even worse, systems offering such languages tend to produce inconsistent outputs, like mixing lexical items from different Portuguese (see for instance the translations shown in Table 6.1). Clearly, in the perspective of delivering high-quality MT to professional post-editors and final users, this problem urges to be fixed.

While machine translation from many to one varieties is intuitively simpler to approach, it is the opposite direction that presents the most relevant problems. First, languages varieties such as dialects might significantly overlap thus making differences among their texts quite subtle (e.g., particular grammatical constructs or lexical divergences like the ones reported in the example). Second, parallel data are not always labeled at the level of language variety, making it hard to develop specific NMT engines. Finally, training data might be very unbalanced among different varieties, due to the population sizes of their respective speakers or for other reasons. This clearly makes it harder to model the lower-resourced varieties [88].

6.2 Related Work

6.2.1 Machine Translation of Language Varieties

Most of the works on translation between and from/to written language varieties involve rule-based transformations, e.g., for European and Brazilian Portuguese [106], Indonesian and Malay [152], Turkish and Crimean Tatar [2]; or phrase-based statistical MT (SMT) systems, e.g., for Croatian, Serbian, and Slovenian [124], Hindi and Urdu [45], or Arabic

[4]In sociolinguistics, a variety is a specific form of language, that may include dialects, registers, styles, and other forms of language, as well as a standard language. See [174] for a more comprehensive introduction.

dialects [65]. Notably, [125] build an unsupervised deciphering model to translate between closely related languages without parallel data. [131] handle mixed Arabic dialect input in MT by using a sentence-level classifier to select the most suitable model from an ensemble of multiple SMT systems. In NMT, however, there have been fewer studies addressing language varieties. It is reported that an RNN model outperforms SMT when translating from Catalan to Spanish [37] and from European to Brazilian Portuguese [38]. [67] propose a technique to augment training data for under-resourced dialects via projecting word embeddings from a resource-rich related language, thus enabling training of dialect-specific NMT systems. The authors generate spoken Levantine-English data from larger Arabic-English corpora and report improvement in BLEU scores compared to a low-resourced NMT model.

6.2.2 Dialect Identification

A large body of research in dialect identification stems from the DSL shared tasks [184, 185, 105, 183]. Currently, the best-performing methods include linear machine learning algorithms such as SVM, naïve Bayes, or logistic regression, which use character and word n-grams as features and are usually combined into ensembles [76]. [165] present the idea of leveraging parallel corpora for language identification: content comparability allows capturing subtle linguistic differences between dialects while avoiding content-related biases. The problem of ambiguous sentences, i.e., those for which it is impossible to decide upon the dialect tag, has been demonstrated for Portuguese by [60] through inspection of disagreement between human annotators.

6.3 Language Variety Aware NMT

Our assumption is to translate from language E (English) into each of two varieties A and B. We assume to have parallel training data $D_{E\to A}$ and $D_{E\to B}$ for each variety as well as unlabeled data $D_{E\to A\cup B}$.

6.3.1 Multilingual Model for Language Variety NMT

As discussed in detail in Chapter 3, a simplified and efficient multilingual NMT approach is to prepend a *language-flag* to the source side of the training segments. This approach has greatly simplified multilingual modeling, by eliminating the need of having separate encoder/decoder network per language pair, as well as maximizing the transfer-learning across the different languages, as discussed in Chapter 5. For our language varieties aware NMT modeling, we follow a similar strategy, by incorporating an artificial token as a unique *variety-flag* on the source (English) side.

6.3.2 Modeling Scenarios

For the sake of experimentation we consider three application scenarios in which a fixed amount of parallel training data E-A and E-B is partitioned in different ways:

- *Supervised*: all sentence pairs are respectively put in $D_{E \to A}$ and $D_{E \to B}$, leaving $D_{E \to A \cup B}$ empty;
- *Unsupervised*: all sentence pairs are jointly put in $D_{E \to A \cup B}$, leaving $D_{E \to A}$ and $D_{E \to B}$ empty;
- *Semi-supervised*: two-thirds of E-A and E-B are, respectively, put in $D_{E \to A}$ and $D_{E \to B}$, and the remaining sentence pairs are put in $D_{E \to A \cup B}$.

6.3.2.1 Supervised and Unsupervised Baselines

For each translation direction we compare three baseline NMT systems. The first system is an unsupervised generic (Gen) system trained on the union of the language varieties training data. Notice that Gen makes no distinction between A and B and uses all data in an unsupervised way. The second is a supervised variety-specific system (Spec) trained on the corresponding language variety training set. The third system (Ada) is obtained by adapting the Gen system to a specific variety (we test this system only on the Portuguese varieties). Adaptation is carried out by simply restarting the training process from the generic model using all the available variety specific training data.

6.3.2.2 Supervised Multilingual NMT

We build on the idea of multilingual NMT (Mul), where one single NMT system is trained on the union of $D_{E\to A}$ and $D_{E\to B}$. Each source sentence both at training and inference time is prepended with the corresponding target language variety label (A or B). Notice that the multilingual architecture leverages the target forcing symbol both as input to the encoder to build its states, and as initial input to the decoder to trigger the first target word.

6.3.2.3 Semi-Supervised Multilingual NMT

We consider here multilingual NMT models that make also use of unlabeled data $D_{E\to A\cup B}$. The first model we propose, named M-U, uses the available data $D_{E\to A}$, $D_{E\to B}$ and $D_{E\to A\cup B}$ as they are, by not specifying any label at training time for entries from $D_{E\to A\cup B}$. The second model, named M-C2, works similarly to Mul, but relying on a language variety identification module (trained on the target data of $D_{E\to A}$ and $D_{E\to B}$) that maps each unlabeled data point either to A or B. The third model, named M-C3, can be seen as an enhancement of M-U, as the unlabeled data is automatically classified into one of three classes: A, B, or $A \cup B$. For the third class, like with M-U, no label is applied on the source sentence.

6.4 Experiments

6.4.1 Dataset and Preprocessing

The experimental setting consists of eight target varieties and English as source. We use publicly available data from the WIT[3] TED corpus [25]. The summary of the partitioned training, dev, and test sets are given in Table 6.2, where Training 2/3 is the labeled portion of the training set used to train the semi-supervised models, while the other 1/3 are either held out as unlabeled (M-U) or classified automatically (M-C2, M-C3). In the preprocessing stages, we tokenize the corpora and remove lines longer than 70 tokens. The Serbian corpus written in Cyrillic is transliterated into Latin script with CyrTranslit[5]. In

[5]https://pypi.org/project/cyrtranslit

	Train	Ratio (%)	Training 2/3	Dev	Test
pt-BR	234K	58.23	156K	1567	1454
pt-EU	168K	47.77	56K	1565	1124
fr-CA	18K	10.26	12K	1608	1012
fr-EU	160K	89.74	106K	1567	1362
hr	110K	54.20	73K	1745	1222
sr	93K	45.80	62K	1725	1214
id	105k	96.71	70K	932	1448
ms	3.6K	3.29	2.4k	1024	738
pt-BR_L	47.2M	64.91	31.4M	1567	1454
pt-EU_L	25.5M	35.10	17M	1565	1124

Table 6.2: *Number of parallel sentences of the TED Talks used for training, development and testing. At the bottom, the large-data set-up which uses the OpenSubtitles (pt-BR_L and pt-PT_L) as additional training set.*

addition, to also run a large-data experiment, we expand the English−European/Brazilian Portuguese data with the corresponding OpenSubtitles2018 datasets from the OPUS corpus. Table 6.2 summarizes the augmented training data, while keeping the same dev and test.

6.4.2 Experimental Settings

We trained all systems using the Transformer model. We use the Adam optimizer [82] with an initial learning rate of 0.2 and a dropout also set to 0.2. A shared source and target vocabulary of size 16k is generated via sub-word segmentation [178]. The choice for the vocabulary size follows the recommendations in [40] regarding training of NMT systems on TED Talks data. Overall we use a uniform setting for all our models, with a 512 embedding dimension and hidden units, and 6 layers of self-attention encoder-decoder network. The training batch size is of 6144 sub-word tokens and the max length after segmentation is set to 70. Following [168] and for a fair comparison, experiments are run for 100k training steps, i.e., in the low-resource settings all models are observed to converge within these steps. Adaptation experiments are run to convergence, which requires roughly half of the steps (i.e., 50k) required to train the generic low-resource

model. On the other hand, large-data systems are trained for up to 800k steps, which also showed to be a convergence point. For the final evaluation we take the best performing checkpoint on the dev set. All models are trained using Tesla V100-pcie-16gb on a single GPU.

6.4.3 Language Variety Identification

To automatically identify the language variety of unlabeled target sentences, we train a fastText model [78], a simple yet efficient linear bag of words classifier. We use both word- and character-level n-grams as features. In the low-resource condition, we train the classifier on the 2/3 portion of the labeled training data. For the large-data experiment, instead, we used a relatively smaller and independent corpus consisting of 3.3 million pt-BR−pt-EU parallel sentences extracted from OpenSubtitles2018 after filtering out identical sentences pairs and sentences occurring (in any of the two varieties) in the NMT training data. Additionally, low-resource training sentences (fr-CA and ms) are randomly over-sampled to mitigate class imbalance.

	pt	sr-hr	fr	id-ms	pt_L
ROC AUC	82.29	88.12	80.99	81.99	52.75

Table 6.3: *Performance of language identification on the low-resource and high-resource (pt_*L*) settings*

For each pair of varieties, we train five base classifiers differing in random initialization. In the M-C2 experiments, prediction is determined based on soft fusion voting, i.e., the final label is the argmax of the sum of class probabilities. Due to class skewness in the evaluation set, we report binary classification performance in terms of ROC AUC [50] instead of accuracy in Table 6.3. For M-C3 models, we handle ambiguous examples using the majority voting scheme: in order for a label to be assigned, its softmax probability should be strictly higher than fifty percents according to the majority of the base classifiers, otherwise no tag is applied. On average, this resulted in <1% of unlabeled sentences for the small data condition, and about 2% of unlabeled sentences for the large data condition.

6.5 Results and Discussion

We run experiments with all the systems introduced in Sec. 6.3, on four pairs of languages varieties. Results are reported in Table 6.4 for the low-resource setting and in Table 6.5 for the large data setting.

6.5.1 Low-Resource Setting

Among the supervised models, which are using all the available training data, the multilingual NMT model Mul outperforms the variety-specific models on all considered directions. Remarkably, the Mul model also outperforms the adapted Ada model on the available translation directions. The unsupervised generic model Gen, that mixes together all the available data, as expected tends to perform better than the supervised specific (Spec) models of the less resourced varieties. Particularly, this improvement is observed for Malay (ms) and Canadian French (fr-CA), which respectively represent the 3.3% and 10% of the overall training data used by their corresponding (Gen) systems. On the contrary, a degradation in the Gen model is observed compared to Spec for European Portuguese (pt-Eu) and Serbian (sr), which represent 47% and 45% of their respective training sets. Even though very low-resourced varieties can benefit from the mix, it is also evident that the Gen model can easily get biased because of the imbalance between the datasets.

In the semi-supervised scenario, we report results with three multilingual systems that integrate the 1/3 of unlabeled data to the training corpus in three different ways: *(i)* without labels (M-U), *(ii)* with automatic labels forcing one of two possible classes (M-C2), *(iii)* with automatic labels of one of the two options or no label in case of low confidence of the classifier (M-C3).

Results show that on average automatic tagging of the unlabeled data is better than leaving them unlabeled, although M-U still remains a better choice than using specialized and generic systems. M-C2 and M-C3 performs on average from very close to better than the best supervised (Mul) method.

If we look at the single language variety (closely related languages), the obtained figures are not showing a coherent picture. In particular, in the Croatian-Serbian and

Supervision Type	Model Type	pt-BR	pt-EU	average
Unsupervised	Gen	↓36.52	↓33.75	35.14
Supervised	Spec	↓35.85	↓35.84	35.85
”	Ada	↓36.54	↓36.59	36.57
”	Mul	**37.86**	**38.42**	**38.14**
Semi-supervised	M-U	↓37.09	37.59	37.34
”	M-C2	**37.70**	**38.35**	**38.03**
”	M-C3	37.59	38.31	37.95
		fr-EU	fr-CA	average
Unsupervised	Gen	**33.91**	↓30.91	32.41
Supervised	Spec	33.52	↓17.13	25.33
”	Mul	33.40	**37.37**	**35.39**
Semi-supervised	M-U	33.28	37.96	35.62
”	M-C2	33.79	↑38.60	36.20
”	M-C3	**↑34.16**	**↑39.30**	**36.73**
		hr	sr	average
Unsupervised	Gen	↓21.71	↓19.20	20.46
Supervised	Spec	↓22.50	↓19.92	21.21
”	Mul	**23.99**	**21.37**	**22.68**
Semi-supervised	M-U	**24.30**	21.53	22.91
”	M-C2	24.14	21.26	22.70
”	M-C3	24.22	**21.97**	**23.10**
		id	ms	average
Unsupervised	Gen	26.56	↓13.86	20.21
Supervised	Spec	26.20	↓2.73	14.47
”	Mul	**26.66**	**15.77**	**21.22**
Semi-supervised	M-U	**26.52**	15.58	21.05
”	M-C2	26.36	**16.31**	**21.34**
”	M-C3	26.40	15.23	20.82

Table 6.4: *BLEU scores of the presented models, trained with unsupervised, supervised and semi-supervised data, from English to Brazilian Portuguese (pt-BR) and European Portuguese (pt-EU), Canadian French (fr-CA) and European French (fr-EU), Croatian (hr) and Serbian (sr), and Indonesian (id) and Malay (ms). Arrows ↓↑ indicate statistically significant differences calculated against* Mul *using bootstrap resampling with* $\alpha = 0.05$ *[85].*

		pt-BR	pt-EU	average
Unsupervised	Gen	↓ 39.78	↓ 36.13	37.96
Supervised	Spec	**41.54**	**40.42**	**40.98**
”	Mul	41.28	40.28	40.78
Semi-supervised	M-U	41.21	39.88	40.55
”	M-C2	41.20	40.02	40.61
”	M-C3	**41.56**	**40.22**	**40.89**

Table 6.5: *BLEU score on the test set of models trained with large-scale data, from English to Brazilian Portuguese (pt-BR) and European Portuguese (pt-EU). Arrows ↓↑ indicate statistically significant differences calculated against the* Mul *model.*

Indonesian-Malay pairs the best resourced language seems to benefit more from keeping the data unlabeled (M-U). Interestingly, even the worst semi-supervised model performs very close or even better than the best supervised model, which suggests the importance of taking advantage of all available data even if they are not labeled.

Focusing on the statistically significant improvements, the best supervised (Mul) is better than the unsupervised (Gen), whereas the best semi-supervised (M-C2 or M-C3) is either comparable or better than the best supervised.

6.5.2 High-Resource Setting

Unlike what observed in the low-resource setting, where Mul outperforms Spec in the supervised scenario, in the large data condition, variety specific models apparently seem the best choice Notice, however, that the supervised multilingual system Mul provides just a slightly lower level of performance with a simpler architecture (one network in place of two). The unsupervised generic model Gen, trained with the mix of the two varieties datasets, performs significantly worse than the other two supervised approaches, this is particularly visible for the pt-EU direction. Very likely, in addition to the ambiguities that arise from naively mixing the data of the two different dialects, there is also a bias effect towards pt-BR which is due to the very unbalanced proportions of data between the two dialects (almost 1:2).

Hence, in the considered high-resource setting, the Spec and Mul models result as best

		pt-BR	pt-EU	average
Unsupervised	M-C2	41.50	**40.21**	40.86
”	M-C3	**41.66**	40.13	**40.90**

Table 6.6: *BLEU scores on the test set by large scale multi-lingual models trained under an unsupervised condition, where all the training data are labeled automatically.*

possible solutions against which comparing our semi-supervised approaches.

In the semi-supervised scenario, the obtained results confirm that our approach of automatically classifying the unlabeled data $D_{E \to A \cup B}$ improves over using the data as they are (M-U). Nevertheless, M-U (semi-supervised) still confirms to perform better than the fully unlabeled Gen (unsupervised) model. In both translation directions, M-C2 and M-C3 get quite close to the performance of the supervised Spec model. In particular, M-C3 shows to outperform the M-C2 model, and even outperforms on average the supervised Mul model. In other words, the semi-supervised model leveraging three-class automatic labels (of $D_{E \to A \cup B}$) seems to perform better than the supervised model with two dialect labels. Besides the comparable BLEU scores, the supervised (Spec and Mul) perform in statistically insignificant way against the best semi-supervised (M-C3), although outperforming the unsupervised (Gen) model. This result raises the question if relabeling all the training data can be a better option than using a combination of manual and automatic labels. This issue is investigated in the next subsection.

Unsupervised Multilingual Models

As discussed in Sec. 6.4.3, the language classifier for the large-data condition is trained on dialect-to-dialect parallel data that does not overlap with the NMT training data. This condition permits hence to investigate a fully unsupervised training condition. In particular, we assume that all the available training data is unlabeled and create automatic language labels for all 47.2M sentences of pt-BR and 25.5M sentences of pt-EU (see Table 6.2). In a similar way as in Table 6.5, we keep the experimental setting of M-C2 and M-C3 models.

Table 6.6 reports the results of the multilingual models trained under the above described unsupervised condition. In comparison with the semi-supervised condition, both

M-C2 and M-C3 show a slight performance improvement. In particular, the three-label M-C3 performs on average slightly better than the two-label M-C2 model. Actually, the little difference is justified by the fact that the classifier used the "third" label only for 6% of the data. Remarkably, despite the relatively low performance of the classifier, average score of the best unsupervised model M-C2 is almost on par with the supervised model Mul.

6.5.3 Translation Examples

English (source)	We offer a considerable number of different refrigerator models. We have also developed a new type of refrigerator. These include American-style side-by-side refrigerators.
pt (Google Translate)	ferecemos um número considerável de modelos diferentes de refrigeradores. Nós também desenvolvemos um novo tipo de geladeira. Estes incluem refrigeradores lado a lado estilo americano.
Low-Resource setting	
pt-BR (M-C2)	Nós oferecemos um número considerável de diferentes modelos de refrigerador. Também desenvolvemos um novo tipo de refrigerador. Eles incluem o estilo americano nas geladeiras lado a lado.
pt-EU (M-C2)	Oferecemos um número considerável de modelos de refrigeração diferentes. Também desenvolvemos um novo tipo de frigorífico. Também desenvolvemos um novo tipo de frigorífico.
High-Resource setting	
pt-BR (Spec)	Oferecemos um número considerável de modelos de geladeira diferentes. Também desenvolvemos um novo tipo de geladeira. Isso inclui o estilo americano lado a lado refrigeradores.
pt-PT (Spec)	Oferecemos um número considerável de modelos de frigorífico diferentes. Também desenvolvemos um novo tipo de frigorífico. Estes incluem frigoríficos americanos lado a lado.
pt-BR (M-C3_L)	Oferecemos um número considerável de diferentes modelos de geladeira. Também desenvolvemos um novo tipo de geladeira. Estes incluem estilo americano lado a lado, geladeiras.
pt-PT (M-C3_L)	Oferecemos um número considerável de diferentes modelos frigoríficos. Também desenvolvemos um novo tipo de frigorífico. Estes incluem estilo americano lado a lado frigoríficos.

Table 6.7: *English to Portuguese translation generated by Google Translate (as of 20th July, 2018) and translations into Brazilian and European Portuguese generated by our semi-supervised multilingual (*M-C2 *and* M-C3_L*) and supervised* Spec *models. For the underlined English terms both their Brazilian and European translation variants are shown. Note, refrigerator is geladeira in pt-BR and frigorífico in pt-EU.*

Finally, in Table 6.7, we show an additional translation example produced by our semi-supervised multilingual models (both under low and high resource conditions) translating into the Portuguese varieties. For comparison we also include output from Google Translate which offers only a generic English-Portuguese direction. In particular, the examples contain the word *refrigerator* that has specific dialect variants. All our variety-specific systems show to generate consistent translations of this term, while Google Translate prefers to use the Brazilian translation variants for these sentences.

6.6 Conclusion

In this chapter, we presented an NMT from English into dialects and related languages (language varieties). We discussed both situations where parallel data is supplied or not supplied with target language/dialect labels. We introduced and compared different neural MT models that can be trained under unsupervised, supervised, and semi-supervised training data regimes. We reported experimental results on the translation from English to four pairs of language varieties with systems trained under low-resource conditions. We show that in the supervised regime, the best performance is achieved by training a multilingual NMT system. For the semi-supervised regime, we compared different automatic labeling strategies that permit to train multilingual neural MT systems with performance comparable to the best supervised NMT system.

Our findings were also confirmed by large scale experiments performed on English to Brazilian and European Portuguese. In this scenario, we have also shown that multilingual NMT fully trained on automatic labels can perform very similarly to its supervised version. The proposed approach can be extended to several NMT application areas, starting from incorporating language varieties in the source side and controllable output generation on the target side. In Chapter 7, we introduce our approach to control the decoder generation of the target language based on a desired output length criteria.

Chapter 7

Controlling the Verbosity of NMT

In this section, we present two approaches to control the verbosity (output length) of a NMT model. In the first approach, we augment the source side with a token representing a specific length-ratio class, i.e. *short, normal, and long*, which at training time corresponds to the observed ratio and at inference time to the desired ratio. In the second approach, inspired by recent work in text summarization [146], we enrich the position encoding used by the transformer model with information representing the position of words with respect to the end of the target string.

We investigate both methods, either in isolation or combined, on two translation directions (En→It and En→De) for which the length of the target is on average longer than the length of the source. We report MT performance results under two training data conditions, small and large, which show limited degradation in BLEU score and n-gram precision as we vary the target length ratio of our models. We also run a manual evaluation which shows for the En→It task a slight quality degradation in exchange of a statistically significant reduction in the average length ratio, from 1.05 to 1.01.

7.1 Problem Statement and Motivation

The seq2seq [9, 151] approach to NMT has shown to improve quality in various translation tasks [14, 66, 96]. While translation quality is normally measured in terms of correct transfer of meaning and of fluency, there are several applications of NMT that would

SRC	It is actually the true integration of the man and the machine.
MT	Es ist tatsächlich die wahre Integration von Mensch und Maschine.
MT*	Es ist die wirkliche Integration von Mensch und Maschine.
SRC	So we thought we would look at this challenge and create an exoskeleton that would help deal with this issue.
MT	Quindi abbiamo pensato di guardare a questa sfida e creare un esoscheletro che potesse aiutare ad affrontare questo problema.
MT*	Pensavamo di guardare a questa sfida e creare un esoscheletro che potesse aiutare a risolvere il problema.

Table 7.1: *German and Italian human and machine translations (MT) are usually longer than their English source (SRC). We investigate enhanced NMT (MT*) that can also generate translations shorter than the source length. Text in red exceeds the length of the source, while underlined words point out the different translation strategy of the enhanced NMT model.*

benefit from optimizing the output length, such as the translation of document elements that have to fit a given layout – e.g. entries of tables or bullet points of a presentation – or subtitles, which have to fit visual constraints and readability goals, as well as speech dubbing, for which the length of the translation should be as close as possible to the length of the original sentence.

Current NMT models do not model explicitly sentence lengths of input and output, and the decoding methods do not allow to specify desired number of tokens to be generated. Instead, they implicitly rely on the observed length of the training examples [111, 145].

Hence, our ultimate goal is to generate translations whose length is not longer than that of the source string (see example in Table 7.1). While generating translations that are just a few words shorter might appear as a simple task, it actually implies good control of the target language. As the reported examples show, the network has to implicitly apply strategies such as choosing shorter rephrasing, avoiding redundant adverbs and adjectives, using different verb tenses, etc.

seq2seq models have been also applied to text summarization [130] to map the relevant information found in a long text into a limited-length summary. Such models have shown promising results by directly controlling the output length [81, 48, 101, 146]. However, differently from MT, text summarization (besides being a monolingual task) is characterized by target sentences that are always much shorter than the corresponding source

sentences. While in MT, the distribution of the relative lengths of source and target depends on the two languages and can significantly vary from one sentence pair to another due to stylistic decisions of the translator and linguistic constraints (*e.g.* idiomatic expressions).

7.2 Existing Approaches

In this section, we review previous work with seq2seq models to control the output length for text summarization, and on the use of tokens to bias the output of NMT.

In text summarization, [81] proposed methods to control output length either by modifying the search process or the seq-to-seq model itself, showing that the latter being more promising. [48] addressed the problem similarly to our token approach, by training the model on data bins of homogeneous output length and conditioning the output on a length token. They reported better performance than [81]. Finally, [146] proposed the extension of the positional encoding of the transformer (cf. Section 2), reporting better performance than [81] and [48].

The use of tokens to condition the output of NMT started with the multilingual models [77, 64], and was then further applied to control the use of the politeness form in English-German NMT [139], in the translation from English into different varieties of the same language [92], for personalizing NMT to user gender and vocabulary [108], and finally to perform NMT across different translation styles [116].

Length Encoding in Summarization:
Recently, an extension of the positional encoding [146] was proposed to model the output length for text summarization. The goal is achieved by computing the distance from every position to the end of the sentence. The new *length encoding* is present only in the decoder network as an additional vector summed to the input embedding. The authors proposed two different variants. The first variant replaces the variable *pos* in the positional embedding of Eq. 2.24 with the difference $len - pos$, where *len* is the sentence length. The second variant *attempts* to model the proportion of the sentence that has been covered at a given position by replacing the constant 10000 in the denominator of Eq. 2.24 with

len.[1] As decoding is performed at the character level, *len* and *pos* are given in number of characters. At training time, *len* is the observed length of the reference summary, while at inference time it is the desired length.

7.3 Controlling the Output Length of NMT

We propose two methods to control the output length in NMT. In the first method we partition the training set in three groups according to the observed length ratio of the reference over the source text. The idea is to let the model learn translation variants by observing them jointly with an extra input token. The second method extends the Transformer positional encoding to give information about the remaining sentence length. With this second method the network can leverage fine-grained information about the sentence length.

7.3.1 Length Token Method

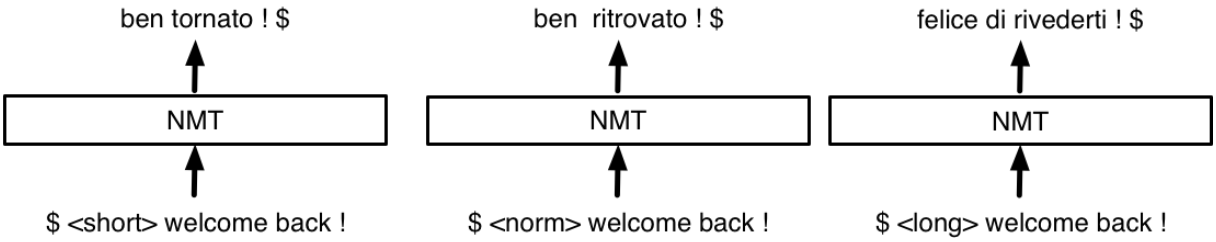

Figure 7.1: *Training NMT with three length ratio classes permits to get outputs of different length at inference time.*

Our first approach to control the length is inspired by *target forcing* in multilingual NMT [77, 64]. We first split the training sentence pairs into three groups according to the target/source length ratio (in terms of characters). Ideally, we want a group where the target is shorter than the source (*short*), one where they are equally-sized (*normal*) and a last group where the target is longer than the source (*long*). In practice, we select two thresholds $t_{\min}$ and $t_{\max}$ according to the length ratio distribution. All the sentence pairs with length ratio between $t_{\min}$ and $t_{\max}$ are in the *normal* group, the ones with ratio below

[1]Notice that the denominator varies with i according to a power function.

$t_{\min}$ in *short* and the remaining in *long*. At training time we prepend a length token to each source sentence according to its group (<short>, <normal>, or <long>), in order to let a single network to discriminate between the groups (see Figure 7.1). At inference time, the length token is used to bias the network to generate a translation that belongs to the desired length group.

7.3.2 Length Encoding Method

Inspired by [146], we use length encoding to provide the network with information about the remaining sentence length during decoding.

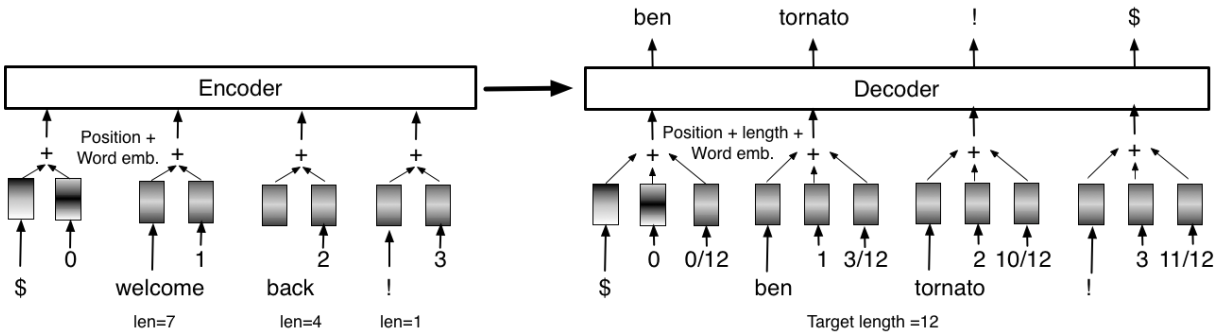

Figure 7.2: *Transformer architecture with decoder input enriched with (relative) length embedding computed according to the desired target string length (12 characters in the example).*

We propose two types of length encoding: *absolute* and *relative*. Let *pos* and *len* be, respectively, a token position and the end of the sequence, both expressed in terms of number characters. Then, the absolute approach encodes the remaining length:

$$\mathrm{LE}_{\mathrm{abs}}(\mathrm{len}, \mathrm{pos}, 2i) = \sin\left(\frac{len - pos}{10000^{\frac{2i}{d_m}}}\right) \tag{7.1}$$

$$\mathrm{LE}_{\mathrm{abs}}(\mathrm{len}, \mathrm{pos}, 2i+1) = \cos\left(\frac{len - pos}{10000^{\frac{2i+1}{d_m}}}\right) \tag{7.2}$$

where $i = 1, \ldots, d_m/2$.

Similarly, the relative difference encodes the relative position to the end. This representation is made consistent with the absolute encoding by quantizing the space of the relative positions into a finite set of N integers:

$$\mathrm{LE}_{\mathrm{rel}}(\mathrm{len}, \mathrm{pos}, 2i) = \sin\left(\frac{q_N(pos/len)}{10000^{\frac{2i}{d_m}}}\right) \tag{7.3}$$

$$\mathrm{LE}_{\mathrm{rel}}(\mathrm{len}, \mathrm{pos}, 2i+1) = \cos\left(\frac{q_N(pos/len)}{10000^{\frac{2i+1}{d_m}}}\right) \tag{7.4}$$

where $q_N : [0,1] \rightarrow \{0,1,..,N\}$ is simply defined as $q_N(x) = \lfloor x \times N \rfloor$. As we are interested in the character length of the target sequence, *len* and *pos* are given in terms of characters, but we represent the sequence as a sequence of BPE-segmented subwords [138]. To solve the ambiguity, *len* is the character length of the sequence, while *pos* is the character count of all the preceding tokens. We prefer a representation based on BPE, unlike [146], as it leads to better translations with less training time [90, 29]. During training, *len* is the observed length of the target sentence, while at inference time it is the length of the source sentence, as it is the length that we aim to match. The process is exemplified in Figure 7.2.

7.3.3 Mixed Length Token and Encoding Approach

We further propose to use the two methods together to combine their strengths. In fact, while the length token acts as a soft constraint to bias NMT to produce short or long translation with respect to the source, actually no length information is given to the network. On the other side, length encoding leverages information about the target length, but it is agnostic of the source length.

7.3.4 Length Control as Fine-Tuning Task

Training an NMT model from scratch is a compute intensive and time consuming task. Alternatively, fine-tuning a pre-trained network shows to improve performance in several NMT scenarios [188, 49, 33, 103, 162]. For our length control approaches, we further propose to use fine-tuning an NMT model with length information, instead of training it from scratch. By adopting a fine-tuning strategy, we specifically aim; *i*) to decouple the performance of the baseline NMT model from that of the additional length information, *ii*) control the level of aggressiveness that can come from the data (length token) and the

model (length encoding), and *iii*) make the approaches versatile to any pre-trained model. More importantly, it will allow to transform any NMT model to an output length aware version, while getting better improvements on the quality of the generated sequences.

7.4 Experiments

7.4.1 Data and Settings

Experiments are run using the English → Italian/German portions of the MuST-C corpus [42], which is extracted from TED talks, using the same train/validation/test split as provided with the corpus (see Table 7.2). As additional data, we use a mix of public and proprietary data for about 16 million sentence pairs for English-Italian (En-It) and 4.4 million WMT14 sentence pairs for the English-German (En-De).

While our main goal is to verify our hypotheses on a large data condition, thus the need to include proprietary data, for the sake of reproducibility in both languages we also provide results with systems only trained on TED Talks (small data condition). When training on large scale data we use Transformer with layer size of 1024, hidden size of 4096 on feed forward layers, 16 heads in the multi-head attention, and 6 layers in both encoder and decoder. When training only on TED talks, we set layer size of 512, hidden size of 2048 for the feed forward layers, multi-head attention with 8 heads and again 6 layers in both encoder and decoder.

Pairs	**Train**	**Dev**	**Test**
En-It (MuST-C)	241,618	1,210	2,574
En-De (MuST-C)	229,703	1,423	2,641
En-De (WMT14)	4,471,497	6,003	3,003

Table 7.2: *Train, validation and test data size in number of examples.*

In all the experiments, we use the Adam [82] optimizer with an initial learning rate of 1×10^{-7} that increases linearly up to 0.001 for 4000 warm-up steps, and decreases afterwards with the inverse square root of the training step. The dropout is set to 0.3 in all layers but the attention, where it is 0.1. The models are trained with label smoothed

Pairs	Set	**short**	**normal**	**long**	**Total**
En-It	train	64185	117589	59844	241618
	dev	247	576	487	1210
	test	599	1200	775	2574
En-De	train	53417	103951	72335	229703
	dev	311	624	488	1423
	test	554	1240	847	2641
Length ratio		$[0, 1]$	$(1, 1.2]$	$(1.2, \infty)$	

Table 7.3: *Train data category after assigning the length tokens (normal, short and long).*

cross-entropy with a smoothing factor of 0.1. Training is performed on 8 Nvidia V100 GPUs, with batches of 4500 tokens per GPU. Gradients are accumulated for 16 batches in each GPU [120]. We select the models for evaluation by applying early stopping based on the validation loss. All texts are tokenized with scripts from the Moses toolkit [87], and then words are segmented with BPE [138] with 32K joint merge rules.

For evaluation we take the best performing checkpoint on the dev set according to the loss. The size of the data clusters used for the length token method and their corresponding target-source length ratios are reported in Table 7.3. The value of N of the relative encoding is set to a small value (5), as in preliminary experiments we observed that a high value (100) produces results similar to the absolute encoding.

7.4.2 Models

We evaluate our Baseline Transformer using two decoding strategies: *i*) a standard beam search inference (standard), and *ii*) beam search with length penalty (penalty) $\alpha = 0.5$ to favor shorter translations [178] as given in Eq. 2.25.

Length token models are evaluated with three strategies that correspond to the tokens prepended to the source test set at a time (short, normal, and long), and reported as Len-Tok. Length encoding (Len-Enc) models are evaluated in a length matching condition, i.e. output length has to match input length. We report the relative (Rel) and absolute (Abs) strategies of the approach as discussed in Section 7.3.2. In the small data condition,

we additionally evaluated how the fine-tuning strategy compares with a model trained from scratch. In the large data condition, we added a setting that combines both the length-token and length-encoding strategies.

7.4.3 Evaluation Criteria

To evaluate all models' performance we compute BLEU on the single-reference test sets of the En-It and En-De pairs. Given the absence of multiple references covering different length ratios, we also report n-gram precision scores (BLEU*), by multiplying the BLEU score by the inverse of the brevity penalty [121]. BLEU* scores is meant to measure to what extent shorter translations are subset of longer translations.

The impact on translation lengths is evaluated with the mean sentence-level length ratios between MT output and source (LR^{src}) and between MT output and reference (LR^{ref}).

7.5 Results

We performed experiments in two conditions: small data and larger data. In the small data condition we only use the MuST-C training set. In the large data condition, a baseline model is first trained on large data, then it is fine-tuned on the MuST-C training set using the proposed methods. Tables 7.4 and 7.5 lists the results for the small and large data conditions. For the two language directions they show BLEU and BLEU* scores, as well as the average length ratios.

7.5.1 Small Data Condition

The baselines generate translations longer than the source sentence side, with a length ratio of 1.05 for Italian and 1.11 for German. Decoding with length penalty (penalty) slightly decreases the length ratios but they are still far from our goal of LR^{src}=1.00.

Fine-Tuning with Length Signal:
A comparison of the models trained from scratch (central portion of Table 7.4) with

Small Data

Pairs		English-Italian				English-German			
Models	**Strategy**	BLEU	BLEU*	LR^{src}	LR^{ref}	**BLEU**	BLEU*	LR^{src}	LR^{ref}
Baseline	standard	32.33	32.33	1.05	1.03	*31.32*	*31.41*	1.11	0.98
	penalty	*32.45*	*32.45*	1.04	1.02	30.80	31.36	*1.09*	0.97
Training from scratch									
	normal	*32.54*	32.54	1.04	1.02	*31.48*	**31.76**	1.12	1.00
Len-Tok	short	31.62	*32.90*	*0.97*	0.95	28.53	31.15	**1.02**	0.90
	long	31.16	31.16	1.10	1.08	30.31	30.31	1.22	1.09
Len-Enc Rel	match	*30.96*	*30.96*	1.03	1.01	29.04	*30.67*	1.06	0.95
Len-Enc Abs	match	30.26	30.26	*1.01*	1.04	27.60	29.58	*1.02*	0.91
Fine-tuning the baseline model									
	normal	32.41	32.41	1.05	1.02	**31.64**	**31.64**	1.12	0.99
Len-Tok	short	**32.67**	**32.80**	**1.01**	0.99	30.12	31.34	*1.07*	0.94
	long	32.00	32.00	1.06	1.04	31.35	31.35	1.15	1.02
Len-Enc Rel	match	*32.10*	*32.10*	1.05	1.03	*30.73*	31.58	1.09	0.97
Len-Enc Abs	match	31.24	31.24	*1.02*	1.01	30.29	*31.65*	*1.07*	0.95

Table 7.4: *Performance of the baseline and models with length information trained from scratch and or by fine-tuning, in terms of BLEU, BLEU*, mean length ratio of the output against the source (LR^{src}) and the reference (LR^{ref}). italics shows the best performing model under each category, while* **bold** *shows the wining strategy.*

their counterparts fine-tuned from the baseline (last portion of Table 7.4) shows that the models in the first group generally generate shorter translations, but of worse quality. Additionally, the results with fine-tuning are not much different from the baseline. Existing models can be enhanced to produce shorter sentences, and little variation is observed in their translation quality.

Length Tokens:

Fine-tuning with Len-Tok (Fourth section in Table 7.4) gives a coarse-grained control over the length, while keeping BLEU scores similar to the baseline or slightly better. Decoding with the token normal leads to translations slightly shorter than the baseline for En-It (LR^{src}=1.05 and LR^{ref}=1.02), while the token small strongly reduces the trans-

lation lengths up to almost the source length (LR^{src}=1.01). In the opposite side, the token long generates longer translations which are slightly worse than the others (32.00). A similar behavior is observed for En-De, where the LR^{src} goes from 1.12 to 1.07 when changing normal with short, and to 1.15 with long. The results with the token long are not interesting for our task and are given only for the sake of completeness.

Length Encoding:
The last section of Table 7.4 lists the results of using length encoding (Len-Enc) relative (Rel) and absolute (Abs). The two encodings lead to different generated lengths, with Abs being always shorter than Rel. Unfortunately, these improvements in the lengths correspond to a significant degradation in translation quality, mostly due to truncated sentences.

7.5.2 Large Data Condition

Our Baselines for the large data condition generate sentences with length ratios over the source comparable to the small data condition (LR^{src} and LR^{ref}), but with better translation quality: 35.46 BLEU points for En-It and 33.96 for En-De. Length penalty slightly reduces the length ratios, which results in a 0.3 BLEU points improvement in Italian and -0.3 in German because of the brevity penalty. In the latter case, the BLEU* is slightly better than the standard baseline output. Also for the large data condition, while the length penalty slightly helps to shorten the translations, its effect is minimal and insufficient for our goal.

Length Token:
In En-It there is no noticeable difference in translation quality between the tokens normal and short, while there is a degradation of ~ 0.7 points when using long. This last result is consistent with the ones observed before. Also in this case the token short does not degrade the BLEU score, and obtains the highest precision BLEU* with 36.22. In En-De we obtain the best results with token normal (34.46), which matches the length distribution of the references. The token short generates much shorter outputs (LR^{src}=1.05), which are also much shorter than the reference ($LR^{ref} = 0.93$). Consequently the BLEU score degrades significantly (30.61), and also the BLEU* is 1 point lower than with the token normal.

Large Data Condition

Pairs		English-Italian				English-German			
Models	**Strategy**	BLEU	BLEU*	LR^{src}	LR^{ref}	**BLEU**	BLEU*	LR^{src}	LR^{ref}
Baseline	standard	35.46	35.46	1.05	1.03	*33.96*	34.06	1.13	0.99
	penalty	**35.75**	35.75	1.04	1.01	33.64	34.19	1.11	0.98
	normal	*35.48*	35.48	1.05	1.02	**34.10**	**34.24**	1.12	1.00
Len-Tok	short	35.39	**36.22**	**1.00**	0.98	30.61	33.27	1.05	0.93
	long	34.71	34.71	1.08	1.05	33.46	33.46	1.21	1.08
Len-Enc Rel	match	*35.18*	*35.18*	**1.01**	0.99	*33.61*	*33.74*	1.11	0.98
Len-Enc Abs	match	33.86	33.86	1.02	1.00	30.79	33.29	1.03	0.92
Tok+Enc Rel	short	34.51	35.91	0.96	0.94	30.08	32.62	**1.01**	0.90
	normal	*35.40*	*35.40*	**1.02**	0.99	*33.41*	*34.09*	1.08	0.96
Tok+Enc Abs	short	33.96	33.96	**1.01**	0.99	29.28	32.28	**1.01**	0.90
	normal	33.90	33.90	**1.01**	1.00	31.19	33.61	1.03	0.92

Table 7.5: *Large scale experiments comparing the baseline, length token, length encoding and their combination.*

Longer translations can be generated with the token long, but they always come at the expense of lower quality.

Length Encoding

For En-It, Len-Enc Rel in Table 7.5 achieves a LR$^{\text{src}}$ of 1.01 with a slight degradation of 0.3 BLEU points over the baseline, while in the case of Abs the degradation is higher (-1.6) and LR$^{\text{src}}$ is similar (1.02). Also in En-De the degradation of Rel over the baseline is only -0.3, but the reduction in terms of LR$^{\text{src}}$ is very small (1.11 vs 1.13). On the other side, Abs produces much shorter translations (1.03 LR$^{\text{src}}$) at the expense of a significantly lower BLEU score (30.79). When computing the BLEU* score, the absolute encoding is only 0.45 points lower than the relative encoding (33.29 vs 33.74), but -0.8 lower than the baseline.

Combining Length Token and Encoding:

So far, we have observed generally good results using the token method and translating with the tokens short and normal. while the length encoding generally produces a more predictable output length, in particular for the absolute variant. In the last experiment,

we combine the two methods in order to have a system that can capture different styles (short, normal, long), as well as explicitly leveraging length information.

The results listed in the last portion of Table 7.5 (Tok+Enc) show that the relative encoding Rel produces better translations than Abs, but again it has less predictability in output length. For instance, in En-It the LR^{src} of Rel is 0.96 with token short and 1.02 with normal, while for En-De it is 1.01 with short and 1.08 with normal. On the other side, the Abs produces LR^{src} of 1.01 with both tokens in En-It and also with short in En-De, and it increases to only 1.03 with normal.

Token	**Scale**	BLEU	BLEU*	LR^{src}	LR^{ref}
	1.00	34.51	**35.91**	0.96	0.94
short	1.10	34.82	35.60	0.98	0.96
	1.20	*35.11*	35.25	**0.99**	0.97
	1.00	**35.40**	35.40	1.02	0.99
normal	0.98	**35.49**	35.49	1.01	0.99
	0.93	**35.46**	*35.67*	**1.00**	0.98

Table 7.6: *Results for En-It with* Tok+Enc Rel *by scaling the target length with different constant factors.*

Controlling Output Length:
In order to achieve LR^{src} as close as possible to 1.0, we set the target length during generation equal to the source length when using the length encoding methods. However, one advantage of length encoding is the possibility to set the target length to modify the average output length. We illustrate this option by using the Tok+Enc Rel system for En-It, and translating with the tokens normal or short and different scaling factors for the target length. The results, listed in Table 7.6, show that we are able to approach an LR^{src} of 1.0 with both tokens and the BLEU score is not affected with token normal (35.45) or improves with token short (35.11).

Discussion:
Length token is an effective approach to generate translations of different lengths, but it does not allow a fine-grained control of the output lengths and its results depend on the partition of the training set into groups, which is a manual process. Length encoding

allows to change the output length, but the two variants have different effects. Absolute encoding is more accurate but generates sentences with missing information. The relative encoding produces better translations than the absolute encoding, but its control over the translation length is more loose. The increased length stability is captured by the standard deviation of the length ratio with the source, which is 0.14 for length tokens, ~ 0.11 for relative encoding and ~ 0.07 for absolute encoding.

The advantage of the combined approach is that it can generate sentences with different style to fit different length groups, and the output length can also be tuned by modifying the target length, while no important quality degradation is observed. Additionally, the standard deviation of the lengths is the same as for the length encoding used.

7.5.3 Human Evaluation and Analysis

	% of Wins	LR^{src}
Baseline	21.96	1.06
Len-Tok	17.99	1.01
P-value	< 0.05	< 0.001

Table 7.7: *Manual evaluation on En-It (large data) ranking translation quality of the baseline (*standard*) and token short translation against the reference translation.*

After manually inspecting the outputs of the best performing models under the large data condition, we decided to run a human evaluation only for the En-It Len-Tok model. As our ultimate goal is to be able to generate shorter translations and as close as possible to the length of the source sentences, we focused the manual evaluation on the Short output class and aimed to verify possible losses in quality with respect to the baseline system. We ran a head-to-head evaluation[2] on the first 10 sentences of each test talk, for a total of 270 sentences, by asking annotators to blindly rank the two system outputs (ties were also permitted) in terms of quality with respect to a reference translation.[3] We

[2]We used crowd-sourcing via figure-eight.com.

[3]Evaluators were asked to tell which version of the sentence was best or if they were equivalent, given that a version is good if both the meaning of the reference is preserved and the grammar is correct.

collected three judgments for each output, from 19 annotators, for a total of 807 scores (one sentence had to be discarded). Inter-annotator agreement measured with Fleiss' kappa was 0.35 (= fair agreement).

Results reported in Table 7.7 confirm the small differences observed in BLEU scores: there are only a 4% more wins for the Baseline and almost 60% of ties. The small degradation in quality of the shorter translations is statistically significant[4] ($p < 0.05$), as well as their difference in length ($p < 0.001$).

EN	And we in the West couldn't understand
MT	*E noi occidentali* non riuscivamo a capire
MT*	*In occidente* non riuscivamo a capire
EN	how much this would restrict freedom of speech
MT	quanto questo avrebbe limitato la libertà
MT*	quanto limitasse la libertà
EN	this is a really extraordinary honor for me
MT	questo è un onore davvero straordinario per me
MT*	per me è un onore straordinario
EN	And this was done
MT	E questo è stato fatto in modo che
MT*	E questo fu fatto in modo che

Table 7.8: *Examples of shorter translation fragments obtained by paraphrasing (italics), drop of words (red), and change of verb tense (underline).*

Notice that the evaluation was quite severe towards the shorter translations, as even small changes of the meaning could affect the ranking. After the manual evaluation, we analyzed sentences in which shorter translations were unanimously judged equal or better than the standard translations. We hence tried to identify the linguistic skills involved in the generation of shorter translations, namely: (i) use of abbreviations, (ii) preference of simple verb tenses over compound tenses, (iii) avoidance of not relevant adjective, adverbs, pronouns and articles, (iv) use of paraphrases. Table 7.8 shows examples of the

[4]We used randomization tests with 15K repetitions [117].

application of the above strategies as found in the test set.

7.6 Conclusion

In this chapter, we have proposed two solutions for the problem of controlling the output length of NMT. A first approach, inspired by multilingual NMT, allows a coarse-grained control over the length and no degradation in translation quality. A second approach, inspired by positional encoding, enables a fine-grained control with only a small error in the token count, but at the cost of lower translation quality. A manual evaluation confirms the translation quality observed with BLEU score.

Our experiments show that both methods can induce the network to generate shorter translations, as well as acquiring interpretable linguistic skills. Moreover, these results confirm the possibility of raising translation quality in a different dimension, here, by generating shorter or normal translations of the source sentence. Given the single reference utilized in this work, it would be worth using more flexible and context-aware evaluations which allow us to account for short translations that are not equivalent to the original but at the same time do not affect the overall meaning of the discourse.

Chapter 8

Conclusions and Future Directions

8.1 Conclusions

In this thesis, we presented our approaches to address the limitations of NMT due to the availability of training data. We focused on two conditions: *i*) *low-resource*, in which language pairs have small parallel data and *ii*) the more difficult *zero-resource*, scenario in which only monolingual resources are available. Our solutions presented in Chapter 3-7 are built on top of multilingual NMT (M-NMT) modeling principles, and leverage self-learning, data augmentation, and transfer-learning techniques.

In Chapter 3, we analyzed the applicability of M-NMT to improve performance in low-resourced translation tasks. Our findings confirmed that a single M-NMT model can perform better than multiple language specific models. Then, we showed that translation performance for low-resource languages in M-NMT can be further improved by adding data covering related language pairs. This technique also provided substantial improvements in the zero-shot translation task. Moreover, we compared two of the prominent architectural choices for NMT, recurrent and self-attention. Our findings show the superiority of the latter in all conditions, namely: single pair, multilingual and zero-shot NMT tasks.

Building on these results, in Chapter 4, we proposed a novel self-learning approach to improve zero-shot translation. Bootstrapping from a baseline M-NMT model, our method utilizes monolingual data for the zero-resource languages in what we called a

train-infer-train loop. The approach consists in leveraging the feedback signals of the primal (source → target) and the dual (target → source) translation directions in an incremental fashion. Experimental results show that our solution can outperform pivot based translation, and performs comparably with supervised NMT models. Our findings show that self-learning can close the gap between translation directions with parallel data and zero-resource language pairs, leading to a *universal multilingual model* even in the absence of parallel training material.

A common transfer-learning approach is to initialize a low-resource (*child*) NMT model with a pre-trained (*parent*) model parameters. In Chapter 5, we proposed a dynamic transfer-learning approach that tailors a parent model to the child model at adaptation time. To this aim, we update the vocabulary entries and the associated parent model parameters, such as embeddings. We evaluated our approach in the *continuous adaptation* from a parent to a single language pair child model and, more interestingly, by *progressively growing* the parent model translation directions into an increasingly multilingual model. Our empirical findings showed a significant improvement over multiple baseline models, and against a static adaptation of a parent model to a new language pair.

In Chapters 6 and 7, we identified interesting problem areas that have commonalities with low-resource translation. We primarily focused on applying the principle of M-NMT the translation into language varieties and styles. In Chapter 6, we proposed an NMT model that can translate into different varieties of a language (e.g. dialects). For instance, for the (English-Portuguese) parallel data, if we classify it into European and Brazilian dialects, unbalanced data distributions can result in a low-resource scenario in at least one of the dialects. Experimental findings show the effectiveness of our approach in multiple, closely-related languages and dialects when compared against strong multilingual and dialect-specific NMT models.

Finally, in Chapter 7, we evaluated our hypothesis of building a single NMT model that can generate outputs with the desired length, such as *short and normal*. We proposed two approaches that aim to preserve the integrity of the original data, instead of splitting and creating separate style-specific models. In the first approach, we introduced length-encoding similar to the transformer model positional encoding. In the second approach, we extended the M-NMT *language-tag* to an output *length-tag*. Our experiments show the latter is much simpler and efficient both with human and automatic evaluation. This

demonstrates the effectiveness of biasing the output of an NMT model to fit specific length constraints, an application that is highly important in headline, subtitle and dubbing oriented translation. More interestingly, the approaches proposed in Chapters 6 and 7 can be applied as a fine-tuning task, to any pre-trained model in a versatile manner.

8.2 Future Directions

To further advance the performance of *low-resource* and *zero-resource* NMT for languages, language varieties and styles, we outline the following aspects as future work:

In a zero-resource scenario, our self-learning approach in Chapter 4, utilized the monolingual data of the zero-resource languages that were paired with the pivot language. A natural extension could be to leverage external monolingual data of the zero-resource languages as in a semi-supervised NMT training [136, 180]. Such a setting potentially allows the zero-shot NMT to incrementally improve by utilizing monolingual material that becomes progressively available. In such a way, we hypothesize that it would be feasible to achieve a true universality of large scale multilingual models [3] with an acceptable translation quality for zero-shot translation directions.

Another direction of research could be to look beyond zero-resource languages that have a common pivot language (such as English). In other words, this would imply applying the approach in settings where the zero-resource languages are visible in the multilingual model paired with distinct pivot languages, so to expand the application of the zero-shot NMT modeling to more zero-resource language scenarios. In Sec. 2.3.7, we indicated *unsupervised NMT* as the alternative for building zero-resource NMT, showing that, it still under-performs in scenarios where the monolingual data is not comparable and the languages are distant [113, 63]. In future work, it will be interesting to combine zero-shot NMT modeling with unsupervised NMT, so to leverage the advantages of both approaches (i.e. multilingual NMT and learning a cross-lingual embeddings space as bootstrapping mechanism).

In our transfer-learning approach, we particularly aimed at tailoring the vocabulary and associated parameters of pre-trained models for a new language direction. Our approach outperformed multiple baseline models and current approaches presented in [113,

173, 1]. An important next step could be to identify other important parent model parameters to update for the best possible transfer-learning. In this direction, [10] proposed to adapt a pre-trained model to new translation directions by simply incorporating language-specific *adapter layers*. Their approach aligns with our *progressive growth* of translation directions in Sec. 5.2. Hence, future works can aim at combing the approach in [10] with our proposal in Chapter 5, to progressively build a universal model. Moreover, [179] showed that adapting a pre-trained model with augmented data of a low-resource language can improve translation performance. Without using such large augmented data, our dynamic adaptation in Sec 5.3 showed comparable or better performance compared to [179]. In future works, dynamic transfer-learning with data augmentation can be seen as a promising strategy to improve the performance of unseen languages when adapted from a pre-trained model.

Another possible research direction is following the approaches we proposed in Chapters 6 and 7, when dealing with translation into language varieties and styles. In principle, this is a problem similar to a M-NMT in a *one-to-many* setting. Following our proposal in Chapter 6, that models a single NMT to translate from a source language into different language varieties, a future direction could be investigating an efficient integration of multiple language varieties in the source side. (i.e. *many-to-one, or many-to-many* translation direction).

Moreover, given the fact that most of the low-resource languages belong to certain language varieties (or in a broader sense, language families), the proposed language-varieties aware NMT can be further extended to progressively grow in translation directions. In other words, our dynamic adaptation approach from Chapter 5 can be utilized to progressively learn new language variety directions. We motivate this, based on our empirical results in (Sec. 3.3), showing that building an M-NMT model based on a language varieties and family criteria is the most effective approach, even for low-resource and zero-resource translation tasks.

Towards controlling the verbosity level of an NMT output (Chapter 7), the priority could be to design more flexible and context-aware evaluations that allow to account for translations (such as *short*) that are not equivalent to the original but at the same time do not affect the overall meaning of the discourse. Such an evaluation platform is necessary since it is missing from the current NMT evaluation settings, both in a high-resource and

low-resource languages. Note that, although evaluation sets with multiple references exist, these references do not account for different styles of the target language (in particular length variability).

Niu and Carpuat [115] proposed an approach to control the *formality level* of NMT output to a particular audience, using synthetic data generated on the fly at time of training. Along this direction, research can focus on extending our output length control approaches to a formality control task as in [115], where the resulting mechanism is not only able to control the translation style in terms of length but the formality level. An ideal application scenario is translating into a different level of complexity while preserving the semantics of the source language.

To conclude, studies show that for the more than $7,000+$ languages that are spoken around the globe [21], very few have enough parallel data to train a usable model [88]. Hence, even with many variants of NMT modeling, the dependency on the availability of a corpus in a parallel format remains a primary challenge for MT research. With the increasing digitization of human activities in social, economic, and political aspects, the absence of working translation models for the majority of languages creates a barrier in the day to day life of billions of people. In conclusion, we would like to highlight the importance of improving NMT performance on zero-resource and low-resource languages to the betterment of people's access to different sources of information.

The contributions and resources derived from the works in this thesis can be accessed in the following repository https://github.com/surafelml/phd-thesis.

Bibliography

[1] Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively multilingual neural machine translation. *arXiv preprint arXiv:1903.00089*, 2019.

[2] Kemal Altintas and İlyas Çiçekli. A Machine Translation System Between a Pair of Closely Related Languages. In *Proceedings of the 17th International Symposium on Computer and Information Sciences (ISCIS 2002)*, pages 192–196, 2002.

[3] Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*, 2019.

[4] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, 2017.

[5] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Unsupervised statistical machine translation. *arXiv preprint arXiv:1809.01272*, 2018.

[6] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. In *Proceedings of the 6th International Conference on Learning Representations*, April 2018.

[7] Amittai Axelrod, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, 2011.

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

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[10] Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. Simple, scalable adaptation for neural machine translation. *arXiv preprint arXiv:1909.08478*, 2019.

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

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

[13] Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus phrase-based machine translation quality: a case study. *arXiv preprint arXiv:1608.04631*, 2016.

[14] Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus phrase-based mt quality: An in-depth analysis on english-german and english-french. *Computer Speech & Language*, 49:52–70, 2018.

[15] Nicola Bertoldi and Marcello Federico. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the fourth workshop on statistical machine translation*, pages 182–189. Association for Computational Linguistics, 2009.

[16] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58, 2014.

[17] Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. Findings of the 2016 conference on machine translation. In

Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers, volume 2, pages 131–198, 2016.

[18] Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, et al. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Workshop on Machine Translation*, 2015.

[19] Ondřej Bojar and Aleš Tamchyna. Improving translation model by monolingual data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 330–336, 2011.

[20] Andrew Donald Booth. *Machine translation of languages, fourteen essays*. Wiley, 1955.

[21] Lyle Campbell. Ethnologue: Languages of the world, 2008.

[22] M Asunción Castano, Francisco Casacuberta, and Enrique Vidal. *Machine translation using neural networks and finite-state models*. Citeseer, 1997.

[23] Isaac Caswell, Ciprian Chelba, and David Grangier. Tagged back-translation. *arXiv preprint arXiv:1906.06442*, 2019.

[24] Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Jan Niehues, Sebastian Stüker, Katsuhito Sudoh, Koichiro Yoshino, and Christian Federmann. Overview of the IWSLT 2017 Evaluation Campaign. In *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan, 2017.

[25] Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy, May 2012.

[26] Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, volume 261, page 268, 2012.

[27] Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. The iwslt 2016 evaluation campaign. *Proc. of IWSLT, Seattle, pp. 14, WA, 2016.*, 2016.

[28] Colin Cherry, George Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. Revisiting character-based neural machine translation with capacity and compression. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4295–4305, 2018.

[29] Colin Cherry, George Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. Revisiting character-based neural machine translation with capacity and compression. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4295–4305, 2018.

[30] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[31] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[32] Jan Chorowski and Navdeep Jaitly. Towards better decoding and language model integration in sequence to sequence models. *arXiv preprint arXiv:1612.02695*, 2016.

[33] Chenhui Chu and Rui Wang. A survey of domain adaptation for neural machine translation. *arXiv preprint arXiv:1806.00258*, 2018.

[34] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[35] Jonathan H Clark, Chris Dyer, Alon Lavie, and Noah A Smith. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 176–181. Association for Computational Linguistics, 2011.

[36] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

[37] Marta R Costa-jussà. Why Catalan-Spanish Neural Machine Translation? Analysis, comparison and combination with standard Rule and Phrase-based technologies. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 55–62, 2017.

[38] Marta R Costa-Jussà, Marcos Zampieri, and Santanu Pal. A Neural Approach to Language Variety Translation. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 275–282, 2018.

[39] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

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

[41] Mattia A. Di Gangi and Marcello Federico. Deep neural machine translation with weakly-recurrent units. In *Proceedings of the 21st Annual Conference of the European Association for Machine Translation (EAMT)*, Alicante, Spain, May 2018. European Association for Machine Translation.

[42] Mattia Antonino Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Minneapolis, MN, USA, June 2019.

[43] Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. Multi-task learning for multiple language translation. In *ACL (1)*, pages 1723–1732, 2015.

[44] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[45] Nadir Durrani, Hassan Sajjad, Alexander Fraser, and Helmut Schmid. Hindi-to-urdu machine translation through transliteration. In *Proceedings of the 48th Annual meeting of the Association for Computational Linguistics*, pages 465–474. Association for Computational Linguistics, 2010.

[46] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

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

[48] Angela Fan, David Grangier, and Michael Auli. Controllable abstractive summarization. *arXiv preprint arXiv:1711.05217*, 2017.

[49] M Amin Farajian, Marco Turchi, Matteo Negri, and Marcello Federico. Multi-domain neural machine translation through unsupervised adaptation. In *Proceedings of the Second Conference on Machine Translation*, pages 127–137, 2017.

[50] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, jun 2006.

[51] Orhan Firat. *CONNECTIONIST MULTI-SEQUENCE MODELLING AND APPLICATIONS TO MULTILINGUAL NEURAL MACHINE TRANSLATION*. arXiv preprint, 2017.

[52] Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv preprint arXiv:1601.01073*, 2016.

[53] Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T Yarman Vural, and Kyunghyun Cho. Zero-resource translation with multi-lingual neural machine translation. *arXiv preprint arXiv:1606.04164*, 2016.

[54] Mikel L Forcada and Ramón P Ñeco. Recursive hetero-associative memories for translation. In *International Work-Conference on Artificial Neural Networks*, pages 453–462. Springer, 1997.

[55] Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027, 2016.

[56] Pablo Gamallo, José Ramom Pichel, and Iñaki Alegria. From language identification to language distance. *Physica A: Statistical Mechanics and its Applications*, 484:152–162, 2017.

[57] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017.

[58] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *arXiv preprint*, 1999.

[59] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[60] Cyril Goutte, Serge Léger, Shervin Malmasi, and Marcos Zampieri. Discriminating Similar Languages: Evaluations and Explorations. In *Proceedings of Language Resources and Evaluation (LREC)*, pages 1800–1807, 2016.

[61] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[62] Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O.K. Li. Universal neural machine translation for extremely low resource languages. In *Proceedings of NAACL-HLT 2018*, pages 344–354, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

[63] Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. Two new evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. *arXiv preprint arXiv:1902.01382*, 2019.

[64] Thanh-Le Ha, Jan Niehues, and Alexander Waibel. Toward multilingual neural machine translation with universal encoder and decoder. *arXiv preprint arXiv:1611.04798*, 2016.

[65] Salima Harrat, Karima Meftouh, and Kamel Smaili. Machine translation for Arabic dialects (survey). *Information Processing & Management*, pages 1–12, aug 2017.

[66] Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. Achieving human parity on automatic chinese to english news translation. *CoRR*, abs/1803.05567, 2018.

[67] Hany Hassan, Mostafa Elaraby, and Ahmed Y Tawfik. Synthetic Data for Neural Machine Translation of Spoken-Dialects. In *Proceedings of the 14th International Workshop on Spoken Language Translation*, 2017.

[68] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[69] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

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

[71] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 328–339, 2018.

[72] John Hutchins. From first conception to first demonstration: the nascent years of machine translation, 1947–1954. a chronology. *Machine Translation*, 12(3):195–252, 1997.

[73] John Hutchins. *Machine translation: A concise history*. Preprint, 2007.

[74] John Hutchins and Evgenii Lovtskii. Petr petrovich troyanskii (1894–1950): A forgotten pioneer of mechanical translation. *Machine translation*, 15(3):187–221, 2000.

[75] W John Hutchins. Machine translation over fifty years. *Histoire epistémologie langage*, 23(1):7–31, 2001.

[76] Tommi Jauhiainen, Marco Lui, Marcos Zampieri, Timothy Baldwin, and Krister Lindén. Automatic Language Identification in Texts: A Survey. In *Preprint*, 2018.

[77] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistic*, 5:339–351, 2017.

[78] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 427–431, 2017.

[79] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, 2013.

[80] Alina Karakanta, Jon Dehdari, and Josef van Genabith. Neural machine translation for low-resource languages without parallel corpora. *Machine Translation*, 32(1):167–189, Jun 2018.

[81] Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. *arXiv preprint arXiv:1609.09552*, 2016.

[82] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[84] Tom Kocmi and Ondřej Bojar. Trivial transfer learning for low-resource neural machine translation. *arXiv preprint arXiv:1809.00357*, 2018.

[85] Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 4, pages 388–395, 2004.

[86] Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

[87] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, et al. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL*, 2007.

[88] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*, 2017.

[89] Julia Kreutzer and Artem Sokolov. Learning to segment inputs for nmt favors character-level processing. In *Proceedings of the 15th International Workshop on Spoken Language Translation*, pages 166–172, 2018.

[90] Julia Kreutzer and Artem Sokolov. Learning to segment inputs for nmt favors character-level processing, 2018.

[91] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*, 2018.

[92] Surafel M Lakew, Aliia Erofeeva, and Marcello Federico. Neural machine translation into language varieties. *arXiv preprint arXiv:1811.01064*, 2018.

[93] Surafel Melaku Lakew, Quintino F Lotito, Marco Turchi, Matteo Negri, and Marcello Federico. Fbk's multilingual neural machine translation system for iwslt 2017. In *14th International Workshop on Spoken Language Translation (IWSLT 2017)*, pages 35–41, 2017.

[94] Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[95] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*, 2018.

[96] Samuel Läubli, Rico Sennrich, and Martin Volk. Has machine translation achieved human parity? A case for document-level evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4796, Brussels, Belgium, 2018.

[97] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[98] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[99] Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378, 2017.

[100] Will Lewis. Haitian creole: How to build and ship an mt engine from scratch in 4 days, 17 hours, & 30 minutes. In *14th Annual conference of the European Association for machine translation*. Citeseer, 2010.

[101] Yizhu Liu, Zhiyi Luo, and Kenny Zhu. Controlling length in abstractive summarization using a convolutional neural network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4110–4119, 2018.

[102] Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*, 2015.

[103] Minh-Thang Luong and Christopher D Manning. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the International Workshop on Spoken Language Translation*, 2015.

[104] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[105] Shervin Malmasi, Marcos Zampieri, Nikola Ljubeši, Preslav Nakov, Ahmed Ali, and Jörg Tiedemann. Discriminating Between Similar Languages and Arabic Dialect Identification: A Report on the Third DSL Shared Task. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 1–14, 2016.

[106] Luis Marujo, Nuno Grazina, Tiago Luis, Wang Ling, Luisa Coheur, and Isabel Trancoso. BP2EP - Adaptation of Brazilian Portuguese texts to European Portuguese. In *Proceedings of the 15th International Conference of the European Association for Machine Translation (EAMT)*, pages 129–136, 2011.

[107] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[108] Paul Michel and Graham Neubig. Extreme adaptation for personalized neural machine translation. *arXiv preprint arXiv:1805.01817*, 2018.

[109] Tomáš Mikolov, Ilya Sutskever, Anoop Deoras, Hai-Son Le, and Stefan Kombrink. Subword language modeling with neural networks. In *Preprint*, 2012.

[110] Robert C. Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, 2010.

[111] Kenton Murray and David Chiang. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, 2018.

[112] Graham Neubig. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint arXiv:1703.01619*, 2017.

[113] Graham Neubig and Junjie Hu. Rapid adaptation of neural machine translation to new languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 875–880. Association for Computational Linguistics, 2018.

[114] Toan Q Nguyen and David Chiang. Transfer learning across low-resource, related languages for neural machine translation. *arXiv preprint arXiv:1708.09803*, 2017.

[115] Xing Niu and Marine Carpuat. Controlling neural machine translation formality with synthetic supervision. *arXiv preprint arXiv:1911.08706*, 2019.

[116] Xing Niu, Sudha Rao, and Marine Carpuat. Multi-task neural models for translating between styles within and across languages. *arXiv preprint arXiv:1806.04357*, 2018.

[117] E.W. Noreen. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley, 1989.

[118] Kemal Oflazer and Ilknur Durgar El-Kahlout. Exploring different representational units in english-to-turkish statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 25–32. Association for Computational Linguistics, 2007.

[119] Bojar Ondřej, Rajen Chatterjee, Federmann Christian, Graham Yvette, Haddow Barry, Huck Matthias, Koehn Philipp, Liu Qun, Logacheva Varvara, Monz Christof, et al. Findings of the 2017 conference on machine translation (wmt17). In *Second Conference on Machine Translation*, pages 169–214. The Association for Computational Linguistics, 2017.

[120] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. *arXiv preprint arXiv:1806.00187*, 2018.

[121] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[122] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

[123] Michael Paul, Marcello Federico, and Sebastian Stüker. Overview of the iwslt 2010 evaluation campaign. In *International Workshop on Spoken Language Translation (IWSLT) 2010*, 2010.

[124] Maja Popović, Mihael Arcan, and Filip Klubička. Language Related Issues for Machine Translation between Closely Related South Slavic Languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pages 43–52, 2016.

[125] Nima Pourdamghani and Kevin Knight. Deciphering Related Languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2503–2508, 2017.

[126] Ofir Press and Lior Wolf. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*, 2016.

[127] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. In *arXiv preprint*, 2017.

[128] Sujith Ravi and Kevin Knight. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21, 2011.

[129] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[130] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[131] Wael Salloum, Heba Elfardy, Linda Alamir-Salloum, Nizar Habash, and Mona Diab. Sentence Level Dialect Identification for Machine Translation System Selection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pages 772–778, 2014.

[132] Helmut Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, 1994.

[133] Rico Sennrich. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 539–549. Association for Computational Linguistics, 2012.

[134] Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. The university of edinburgh's neural mt systems for wmt17. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 389–399, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[135] Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, et al. Nematus: a toolkit for neural machine translation. *arXiv preprint arXiv:1703.04357*, 2017.

[136] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

[137] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[138] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[139] Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, 2016.

[140] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725. Association for Computational Linguistics (ACL), 8 2016.

[141] Rico Sennrich, Martin Volk, and Gerold Schneider. Exploiting Synergies Between Open Resources for German Dependency Parsing, POS-tagging, and Morphological Analysis. In *Proceedings of Recent Advances in Natural Language Processing*, pages 601–609, 2013.

[142] Rico Sennrich and Biao Zhang. Revisiting low-resource neural machine translation: A case study. *arXiv preprint arXiv:1905.11901*, 2019.

[143] Claude E Shannon and Warren Weaver. *The mathematical theory of communication*. University of Illinois press, 1998.

[144] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[145] Xing Shi, Kevin Knight, and Deniz Yuret. Why neural translations are the right length. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2278–2282, 2016.

[146] Takase Sho and Okazaki Naoaki. Positional encoding to control output sequence length. In *Proceedings of the HLT-NAACL 2019*, 2019.

[147] H.T. Siegelmann and E.D. Sontag. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50(1):132 – 150, 1995.

[148] Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, Boston, US-MA, August 2006.

[149] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200, 2006.

[150] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

[151] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[152] Tien-Ping Tan, Sang-Seong Goh, and Yen-Min Khaw. A Malay Dialect Translation and Synthesis System: Proposal and Preliminary System. In *2012 International Conference on Asian Language Processing*, pages 109–112. IEEE, nov 2012.

[153] Odlin. Terence. Language transfer-cross-linguistic influence in language learning. *Cambridge University Press. Cambridge Books Online.*, page 222, June 1989.

[154] **Lakew, Surafel Melaku**, Mauro Cettolo, and Marcello Federico. "A Comparison of Transformer and Recurrent Neural Networks on Multilingual Neural Machine Translation. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, New Mexico, USA, 2018.

[155] **Lakew, Surafel Melaku**, Mattia Di Gangi, and Marcello Federico. "Controlling the Output Length of Neural Machine Translation". In *16th International Workshop on Spoken Language Translation (IWSLT)*, Hong Kong, 2019.

[156] **Lakew, Surafel Melaku**, Mattia Antonino Di Gangi, and Marcello Federico. "Multilingual Neural Machine Translation for Low Resource Languages". In *Proceedings of the 4th Italian Conference on Computational Linguistics (CLiC-IT)*, Rome, Italy, 2017.

[157] **Lakew, Surafel Melaku**, Aliia Erofeeva, and Marcello Federico. "Neural Machine Translation into Language Varieties". In *Proceedings of the Third Conference on Machine Translation: Research Papers (WMT)*, Brussels, Belgium, 2018.

[158] **Lakew, Surafel Melaku**, Aliia Erofeeva, Matteo Negri, Marcello Federico, and Marco Turchi. "Transfer Learning in Multilingual Neural Machine Translation with Dynamic Vocabulary". In *15th International Workshop on Spoken Language Translation (IWSLT)*, Bruges, Belgium, 2018.

[159] **Lakew, Surafel Melaku**, Marcello Federico, Matteo Negri, and Marco Turchi. "Multilingual Neural Machine Translation for Low Resource Languages". In *Italian Journal of Computational Linguistics (IJCoL)*, Rome, Italy, 2018.

[160] **Lakew, Surafel Melaku**, Alina Karakanta, Marcello Federico, Matteo Negri, and Marco Turchi. "Adapting Multilingual Neural Machine Translation to Unseen Languages". In *16th International Workshop on Spoken Language Translation (IWSLT)*, Hong Kong, 2019.

[161] **Lakew, Surafel Melaku**, Quintino F Lotito, Negri Matteo, Turchi Marco, and Federico Marcello. "Improving Zero-Shot Translation of Low-Resource Languages".

In *14th International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan, 2017.

[162] Brian Thompson, Huda Khayrallah, Antonios Anastasopoulos, Arya D McCarthy, Kevin Duh, Rebecca Marvin, Paul McNamee, Jeremy Gwinnup, Tim Anderson, and Philipp Koehn. Freezing subnetworks to analyze domain adaptation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 124–132, 2018.

[163] Jörg Tiedemann. Parallel data, tools and interfaces in opus. In *Arxiv preprint*, 2012.

[164] Jörg Tiedemann. Emerging language spaces learned from massively multilingual corpora. *arXiv preprint arXiv:1802.00273*, 2018.

[165] Jörg Tiedemann and Nikola Ljubeši. Efficient Discrimination Between Closely Related Languages. In *Proceedings of COLING 2012: Technical Papers*, pages 2619–2634, 2012.

[166] Antonio Toral and Víctor M Sánchez-Cartagena. A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions. *arXiv preprint arXiv:1701.02901*, 2017.

[167] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*, 2016.

[168] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[169] Bernard Vauquois. A survey of formal grammars and algorithms for recognition and transformation in mechanical translation. In *IFIP Congress*, 1968.

[170] Lyan Verwimp, Joris Pelemans, Hugo Van hamme, and Patrick Wambacq. Character-word lstm language models. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 417–427, 2017.

[171] David Vilar, Jia Xu, Luis Fernando d'Haro, and Hermann Ney. Error analysis of statistical machine translation output. In *Proceedings of LREC*, pages 697–702, 2006.

[172] Stefan Wager, Sida Wang, and Percy S Liang. Dropout training as adaptive regularization. In *Advances in neural information processing systems*, pages 351–359, 2013.

[173] Xinyi Wang, Hieu Pham, Philip Arthur, and Graham Neubig. Multilingual neural machine translation with soft decoupled encoding. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[174] Ronald Wardhaugh. *An Introduction to Sociolinguistcs*. Blackwell Publishing, 2006.

[175] Marlies van der Wee, Arianna Bisazza, and Christof Monz. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410. Association for Computational Linguistics, 2017.

[176] Paul J Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

[177] Hua Wu and Haifeng Wang. Pivot language approach for phrase-based statistical machine translation. *Machine Translation*, 21(3):165–181, 2007.

[178] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[179] Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. Generalized data augmentation for low-resource translation. *arXiv preprint arXiv:1906.03785*, 2019.

[180] Yingce Xia, Di He, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. *CoRR*, abs/1611.00179, 2016.

[181] Wenduan Xu, Michael Auli, and Stephen Clark. Ccg supertagging with a recurrent neural network. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 250–255, 2015.

[182] Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. Unsupervised neural machine translation with weight sharing. *arXiv preprint arXiv:1804.09057*, 2018.

[183] Marcos Zampieri, Shervin Malmasi, Nikola Ljubeši, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. Findings of the VarDial Evaluation Campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 1–15, 2017.

[184] Marcos Zampieri, Liling Tan, Nikola Ljubešić, Jörg Tiedemann, and Nikola Ljubeˇ. A Report on the DSL Shared Task 2014. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 58–67, 2014.

[185] Marcos Zampieri, Liling Tan, Nikola Ljubešić, Jörg Tiedemann, and Preslav Nakov. Overview of the DSL Shared Task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pages 1–9, 2015.

[186] Barret Zoph and Kevin Knight. Multi-source neural translation. *arXiv preprint arXiv:1601.00710*, 2016.

[187] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575. Association for Computational Linguistics, 2016.

[188] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. *arXiv preprint arXiv:1604.02201*, 2016.