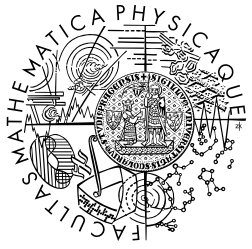


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

DOCTORAL THESIS

Tom Kocmi

Exploring Benefits of Transfer Learning in Neural Machine Translation

Institute of Formal and Applied Linguistics

Supervisor of the doctoral thesis:	doc. RNDr. Ondřej Bojar, Ph.D.
Study programme:	Computer Science
Specialization:	Mathematical Linguistics

Prague 2019

I declare that I carried out this doctoral thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

Prague, September 30, 2019 Tom Kocmi

Title:	Exploring Benefits of Transfer Learning in Neural Machine Translation
Author:	Tom Kocmi
Department:	Institute of Formal and Applied Linguistics
Supervisor:	doc. RNDr. Ondřej Bojar, Ph.D., Institute of Formal and Applied Linguistics
Keywords:	transfer learning, machine translation, deep neural networks, low-resource languages

Abstract:
Neural machine translation is known to require large numbers of parallel training sentences, which generally prevent it from excelling on low-resource language pairs. This thesis explores the use of cross-lingual transfer learning on neural networks as a way of solving the problem with the lack of resources. We propose several transfer learning approaches to reuse a model pretrained on a high-resource language pair. We pay particular attention to the simplicity of the techniques. We study two scenarios: (a) when we reuse the high-resource model without any prior modifications to its training process and (b) when we can prepare the first-stage high-resource model for transfer learning in advance. For the former scenario, we present a proof-of-concept method by reusing a model trained by other researchers. In the latter scenario, we present a method which reaches even larger improvements in translation performance. Apart from proposed techniques, we focus on an in-depth analysis of transfer learning techniques and try to shed some light on transfer learning improvements. We show how our techniques address specific problems of low-resource languages and are suitable even in high-resource transfer learning. We evaluate the potential drawbacks and behavior by studying transfer learning in various situations, for example, under artificially damaged training corpora, or with fixed various model parts.

Acknowledgements

I will always remember my doctoral years as an intensive period of my life, full of both amazing and tough experiences. I could have never finished my thesis without the inspiration and support of many excellent people around me.

First of all, I would like to express my thanks to my supervisor Ondřej Bojar for his excellent guidance, exceptional encouragement, and his willingness to help. Getting to know him is one of my biggest fortunes and the most precious part of my PhD study. I cannot express my gratitude enough for his support and extraordinary care over the years.

I am very grateful to all my friends and colleagues at the Institute of Formal and Applied Linguistics, for being such friendly people with whom it is such a pleasure to work. I cannot forget all the invaluable time we spent discussing research and life, especially during our after-lunch coffee expeditions.

Special thanks go to all the inhabitants of office 423. I wouldn't have made it without your encouragement and support.

With special regards, I would like to thank Martin Popel, Jindřich Libovický, prof. Martin Holeňa, Shadi Saleh, Barbora Chattová, Miklós Danka, Tomáš Musil, and others who read and commented on earlier versions of this thesis.

Many thanks to my dear friends, and to people from StopTime for keeping me sane during my studies. Especially, I thank them for teaching me not to forget about fun in my life. Special thanks to Quido for his never-ending supplies of coffee and much more.

Finally, I would like to express my wholehearted thanks to my mother and sister for the endless patience, love, and encouragement they provided me throughout my entire life. Special thanks to my grandfather František, who taught me to be always curious, which was the essence that inspired me to become a scientist.

The work on this thesis was supported by the Charles University Grant Agency (GAUK 8502/2016, SVV 260 333, SVV 260 453), the Czech Science Foundation (18-24210S), and projects 825303 (Bergamot), H2020-ICT-2014-1-645442 (QT21) of the European Union. This work has been using language resources and tools developed, stored, and/or distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071).

To my amazing mom.

Contents

1 Introduction

The Babel fish is small, yellow and leech-like, and probably the oddest thing in the Universe. If you stick a Babel fish in your ear, you can instantly understand anything said to you in any form of language.
–The Hitchhiker's Guide to the Galaxy. Douglas Adams

With the spread of technology, people around the world are becoming more connected than ever before, and the need for seamless communication and understanding becomes crucial. According to Simons (2018), there are 7097 living languages in the world. However, most of the language pairs have at most hundreds to thousands of parallel sentences with a limited set of paired languages. This lack of data is a severe problem for the training of suitable Machine Translation (MT) systems because both Statistical Machine Translation (SMT) and Neural Machine Translation (NMT) are data demanding machine learning approaches.

Before my doctoral study, the primary approach for MT was Phrase-Based Machine Translation (PBMT) (Koehn et al., 2003; Bojar et al., 2015). However, a complete paradigm shift with the rise of NMT approaches a few years ago (Bojar et al., 2016, 2017).

Currently, NMT is a common approach to automatic translation, and according to Hassan et al. (2018); Bojar et al. (2018) it starts to reach human parity in some language pairs. However, it is only valid for high-resource language pairs where we have plenty of available data – usually dozens of millions of parallel sentences. The performance with low amounts of data can be dramatically reduced up to the point where PBMT systems outperform NMT (Koehn and Knowles, 2017). To some extent, the problem can be mitigated if the NMT model is scaled down accordingly (Sennrich and Zhang, 2019), but it does not resolve the data demands.

The goal of this thesis is to study transfer learning techniques to improve the performance of NMT, especially for translating low-resource language pairs. In general, transfer learning refers to the use of vaguely related training data

to improve the performance at the desired task. For instance, in NMT, transfer learning reuses the model, or its parts, trained for one language pair to improve the performance in a different language pair.

1.1 Contributions

The main contributions of this work are the proposed transfer learning techniques with a broad analysis. We show several approaches on how to improve the performance of (not only) low-resource language pairs with a model trained for a different high-resource language pair. During our analysis, we show that:

- Transfer learning works for both low and high-resource language pairs and achieves better performance than training from random initialization.
- Transfer learning in NMT does not have negative effects known in other fields and can be used as an initialization method for NMT experiments.
- We show that the quantity of parallel corpus plays a more important role in transfer learning than relatedness of language pairs.
- We observe that transfer learning works as a better initialization technique and improves performance even when no language is shared between both models.

Apart from the main contributions, we also describe several other research ideas, starting with our contribution to the development of Czech–English parallel corpora (Bojar et al., 2016a), experiments with pretrained word embeddings (Kocmi and Bojar, 2017c), word embeddings with subword information (Kocmi and Bojar, 2016), a neural language identification tool (Kocmi and Bojar, 2017b). We also contributed to the implementation of a research sequence-to-sequence framework Neural Monkey (Helcl et al., 2018).

During my doctoral study, we investigated the use of curriculum learning (Kocmi and Bojar, 2017a), helped to prepare a Neural Training Shared Task at WMT 2017 (Bojar et al., 2017), and developed a neural abstractive summarization tool (Straka et al., 2018). Furthermore, we participated in several shared tasks (Bojar et al., 2016b; Sudarikov et al., 2017; Kocmi et al., 2017, 2018a,b,c; Kocmi and Bojar, 2019b).

The complete list of publications that I co-authored during my doctoral study can be found in the *List of Publications* on page 136. All publications went through the peer-review process. The only exception is Kocmi and Bojar (2019a) that is not yet published.

1.2 Structure of the Thesis

In the following chapters, we describe our approach to low-resource transfer learning in NMT. This thesis consists of two main parts: the description and evaluation of our transfer learning approaches and a broad analysis of the approach. The thesis includes results and text snippets from our published

works. Short textual parts are reused only from works without other co-authors except for my supervisor. Furthermore, the works used in this thesis are stated at the beginning of individual sections.

- In Chapter 2, we introduce the definition of low-resource languages and describe language resource categories for MT. We describe Czech–English corpus (Bojar et al., 2016a), show the problem with noisy parallel sentences, and present our tool for neural language identification (Kocmi and Bojar, 2017b). Lastly, we describe the standard approaches to the evaluation of machine translation.
- In Chapter 3, we describe NMT architectures with a focus on word embeddings and segmentation of words. We present our approach to word embeddings that includes subword information in word representation (Kocmi and Bojar, 2016).
- In Chapter 4, we present one of the central parts of this thesis: transfer learning for NMT. We describe two scenarios, which differ in assumed conditions on the transferred model. We propose several approaches for both scenarios and evaluate them. This chapter is based mostly on our two papers: Kocmi and Bojar (2018) and Kocmi and Bojar (2019b).
- Equally important is Chapter 5, where we analyze the gains by transfer learning and shed some light on the understanding of the underlying mechanisms.
- We conclude the thesis in Chapter 6.

2 Background

In this chapter, we discuss the definition of low-resource languages, sources, and quality of parallel corpora in Section 2.1. In Section 2.2, we describe our language identification tool that is going to be used later in transfer learning analysis. Then we describe the training data used throughout our work in Section 2.3. Lastly, we explain how the MT is evaluated in Section 2.4.

2.1 Language Resources

Understanding and collecting the available resources is a crucial step towards training NMT systems. In general, we can classify resources based on their domain, their size, and their quality. Another criterion is whether they are monolingual or parallel.

The availability of parallel sentences is crucial for NMT. It is expensive and hard to obtain a large number of parallel sentences. On the other hand, it is easier to obtain a monolingual data by crawling the Internet or from various online sources. The amount of available monolingual data in the target language typically far exceeds the number of parallel sentences. Thus researchers have been trying to utilize monolingual data for MT and we investigate it in Section 5.6.

A comparably important criterion to the size of the parallel corpus is its domain and quality. It is well-known that domain-specific training data are better for the final performance than some general data. The same holds for the quality where a large, noisy training set can notably hinder the performance of an NMT system (e.g. survey by Chu and Wang, 2018).

Throughout the thesis, we use standard shortcuts "k" for thousand and "M" for million. In the case of a parallel corpus, we often use "sentences" as a shortcut for "sentence pairs".

Whenever we talk about a language pair, we use a dash between the languages, e.g. Czech–English. On the other hand, whenever we talk about actual

translation direction, we use an arrow to specify a direction from which language to which we translate, e.g. English→Czech.

In this section, we introduce a definition of an MT domain, followed by an examination of the quality of parallel corpus and approaches for dataset cleaning.

2.1.1 Definition of Domain in Machine Translation

The definition of a domain varies among papers. In general, it is considered any set of instances from a dataset containing a common feature. In most of the papers concerning domain adaptation, the authors define the domain as the source of the dataset. This domain is closely related to the topic or genre of the documents (Hildebrand et al., 2005; Chu et al., 2017; Servan et al., 2016). Examples of such domains are subtitles, literature, news, medical reports, patents, IT, and many more. All of them vary in the used vocabulary, style of writing, and content.

Another feature is the formality or informality of the document, which is closely related to honorifics in languages like Czech, German, or Japanese (Sennrich et al., 2016c). It is a way of encoding the relative social status of speakers to the readers, and for many styles, like official documents, it is an important feature determining the quality of the translation.

Further, we can distinguish documents based on sentiment. The sentiment tone of a text can change with machine translation (Glorot et al., 2011; Mohammad et al., 2016) mainly because of language differences and ambiguity. Another issue with sentiment is that the same information can be written from a positive, neutral, or negative stance. We can go even further and distinguish documents based on the writing style of the author or expected style preference of the reader, as of formality of a speech, specialized vocabulary, or dialects (Jeblee et al., 2014).

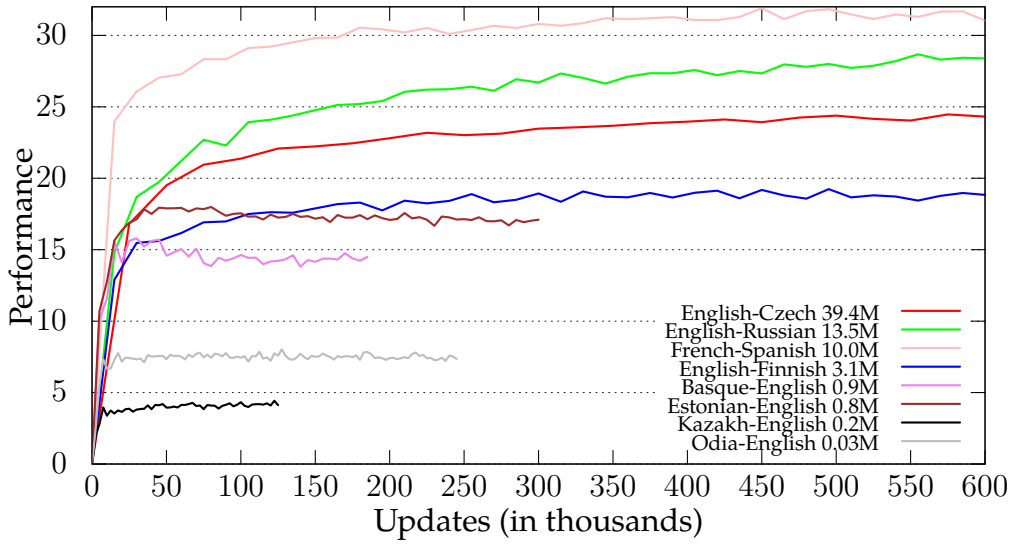
2.1.2 Definition of Low-Resource Languages

Recently, rapid development of NMT systems led to the claims that the human parity has been reached on high-resource language pairs like Chinese–English (Hassan et al., 2018) or Czech–English (Bojar et al., 2018). However, NMT systems tend to be very data-hungry. Koehn and Knowles (2017) have shown that in the low-resource scenarios, NMT lag behind PBMT approaches. This problem led to the rise of attention in low-resource NMT in recent years.

A precise definition of language pairs that count as low-resource is a research question itself. One must consider all aspects of available language resources as well as the language itself.

One of the aspects is the domain of the parallel corpus. Having a large amount of domain-specific parallel sentences can be considered high-resource in the given domain, but low-resource in the general domain, where the performance can be terrible. For example, one common source of parallel sentences for low-resource languages is the Bible, which is translated into hundreds of languages (Christodouloupoulos and Steedman, 2015). However, it is a highly domain and style specific text.

Figure 2.1: Learning curves for various language pairs with various sizes of parallel corpus. We can see that language pairs with less than 1M data quickly flatten out or even start overfitting as in the case of Basque→English.



Highly-inflected languages further complicate the definition of low-resource by presenting a notable sparsity problem with the various forms of inflected words and therefore requiring more parallel sentences to reach comparable performance as the translation of less inflected languages (Denkowski and Neubig, 2017).

Gu et al. (2018) define the *extremely low-resource scenario* by the minimal amount of data needed for NMT to obtain a reasonable translation quality. They showed that extremely low-resource scenario could be considered up to the 13-28k parallel sentences for English→Romanian translation.

In recent years, researchers have organized several machine translation shared tasks in the low-resource scenario. Niehues et al. (2018) introduced a task on Basque→English low-resource translation with an available in-domain corpus of 6k sentence pairs and 940k out-of-domain sentence pairs. The low-resource translation tasks in Workshop on Statistical Machine Translation (WMT) 2018 (Bojar et al., 2018) have been Estonian–English with 880k and Turkish–English with 208k parallel sentences. This year in WMT (Bojar et al., 2019), the low-resource language was Gujarati–English with 170k parallel sentences and Kazakh–English with 220k parallel sentences.

Notably, Koehn and Knowles (2017) found out that NMT outperforms SMT when more than 24.1M words are available, i.e., approximately 1M Spanish–English parallel sentences. However, we need to add that Sennrich and Zhang (2019) recently revisited the training condition for low-resource and showed that current systems outperform SMT even in the low-resource scenario if careful training is performed.

Thus a usual definition is that language pairs with less than a million training pairs are deemed low-resource.

Another point of view, which can be used to delineate low-resources is behavior during training. Figure 2.1 shows so-called learning curves, i.e., the performance of a given system on a development set (also called held-out set) throughout the training. When the curves bend down (e.g. the performance starts decreasing), it is an indication that the model is overfitted, usually by memorizing training sentences.

During our experiments, we noticed that low-resource languages often overfit or flatten out within first 50-100k of weight updates, which in our setting is roughly half a day of training on a single GPU. In contrast, the higher-resource language pairs rarely overfit, usually flatten out after several hundred thousand steps, see Figure 2.1 for an illustration.

We should also note that a language pair considered low-resource today might not be considered low-resource in the future: either due to the newly available data but also due to the improvements in NMT training techniques.

2.1.3 Resource Quality

Language resources for machine translation come in varying degrees of quality. On the one extreme, there are proper translations made by professional translators, which result in parallel corpora with more aligned translations and adherence to sentence-to-sentence alignment. On the other extreme sentences automatically extracted from noisy crawled web pages, which are often miss-aligned, contain non-word parts, and even sentences in a wrong language.

Another issue with parallel corpora is the phenomenon called translationese (Gellerstam, 1986). A text translated from one language to the second one has different linguistic properties than text written in the second language originally. According to Baker et al. (1993), translated texts are often:

- Simplified – when translators subconsciously simplify the message.
- Normalized – to conform the typical features of target languages up until a point of a slight change in meaning.
- Explicitated – the notion that structures of text are explained in more detail due to the rarity of the phenomenon in the target language, for example, explaining abbreviations. It is, to some extent, an inverse to the simplification.

Stymne (2017) evaluated the effect of translationese on the MT systems and found out that the translation direction indeed influences the final quality. However, as the authors mention, their study would need to be evaluated over a larger sample of language pairs and MT systems to be more reliable.

2.1.4 Corpus Cleaning

Collecting training data is the first step needed in order to start training machine translation models. For low-resource languages, crawled data are often one of the largest sources of parallel sentences. Unfortunately, the resources based on crawled data are usually noisy. In order to use the crawled data, we need to clean them first.

There is a whole field for filtering parallel corpora. Koehn et al. (2018) prepared a shared task intending to study various techniques of filtering parallel corpus for NMT to improve its performance by reducing noisy data. Filtering usually consists of pre-filtering rules, like removing non-word tokens, various tags or sentences with mismatched lengths. More advanced techniques rely on various scoring functions.

Furthermore, we can remove short segments of up to a few tokens. These sentences are often relics of misaligned pairs. A standard is to remove sentences with less than five tokens (Koehn et al., 2018).

In contrast, we can also think about removing long segments or breaking them to shorter ones. For practical reasons of batch training on GPU, sentences within one batch are padded to a fixed length according to the longest sentence in the batch. Thus in order to increase the batch size, we can remove very long sentences from the corpus, because they result into large padding of other sentences in the same batch. However, removing long sentences should be done only in the case if there are very few of them compared to the size of the whole corpus so that their removal will not affect the overall number of parallel sentences. In our papers, we followed the recommendation of Popel and Bojar (2018) and set the threshold to 100 or 150 tokens as we have not witnessed any change in the performance, but it allows us to increase the batch size, which is beneficial for the training (Popel and Bojar, 2018).

Lastly, whenever we are dealing with a corpus that was automatically collected, we may want to check all sentences by automatic language identification tool in order to remove sentences that are in a different language. Although this step can remove correct sentences due to errors done by the automatic language identification, it is usually beneficial since it removes a part of noisy sentences.

We have also contributed to the field of language identification by developing a neural language identification tool called Language Identification by Neural Networks (LanideNN). We describe the tool in the following section.

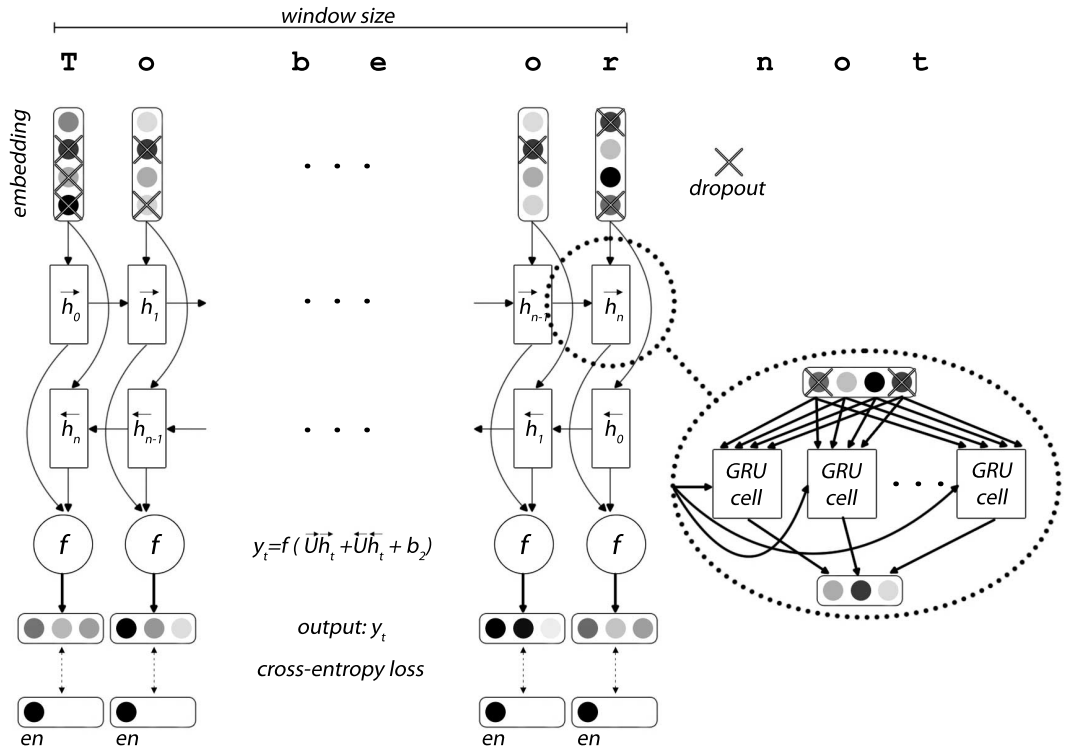
2.2 LanideNN: Language Identification Tool

In language identification, we want to determine the language of some input text automatically. Monolingual language identification assumes that the given document is written in a single language. In multilingual language identification, the document is usually in two or three languages, and we only want their names. In the task of code-switching identification, where the speaker of one language uses words from a different second language, we need to tag individual words or phrases by intended language.

Techniques of language identification can rely on handcrafted rules, usually of high precision but low coverage, or data-driven methods that learn to identify languages based on sample texts of sufficient quantity (Cavnar et al., 1994; Yang and Liang, 2010; Carter et al., 2013).

In Kocmi and Bojar (2017b), we have developed a neural language identification tool. It focuses on multilingual identification as well as language identification from short segments. This section includes results and textual parts of our paper. We use this tool for analysis of transfer learning behavior in Section 5.1.1.

Figure 2.2: Illustration of our LanideNN model architecture. The input on top is processed through the network to get assigned language label at bottom based on argument maximum.



To our knowledge, and also according to the survey by Garg et al. (2014), we have been the first[1] who approached the language identification with neural networks, and further, we reached the state-of-the-art in multilingual language identification in the year 2017 (Kocmi and Bojar, 2017b).

The method we propose is designed for short texts without relying on sentence or document boundaries. If documents are known and if they can be assumed to be monolingual, this additional knowledge should not be neglected. However, for the long term, we aim at streamlined processing of noisy data genuinely appearing in multilingual environments. For instance, our method could support the study of code-switching in e-mails or other forms of conversation, or to analyze various online media such as Twitter, for example, see Montes-Alcalá (2007).

The model is trained on a character sliding window of input texts. It takes individual source characters as input and provides a language label for each of them. Whenever we need to recognize the language of a document, we take the language assigned by our model to the majority of characters. Our model

[1]One exception in using Neural Network (NN) for language identification task before our work is Al-Dubaee et al. (2010), who combine a feed-forward network classifier with wavelet transforms of feature vectors to identify English and Arabic from the Unicode representation of words, sentences or whole documents. The benefit of NN in this setting is not very clear to us because the writing scripts of studied languages can alone distinguish English from Arabic.

Table 2.1: Results of monolingual language identification. Entries marked with * are accuracies reported by Lui and Baldwin (2012), the rest are our measurements.

System	Supported languages	EuroGov	TCL	Wikipedia
* LangDetect	53	–	.818	.867
* TextCat	75	.941	.605	.706
* CLD	64	.983	.732	.831
Langid.py	97	**.987**	.931	**.913**
CLD2	83	.979	.837	.854
Our model	**132**	.977	**.954**	.893

operates on a window of 200 characters of input text, i.e. individual characters, encoded in Unicode. The model classifies each character separately but quickly learns to classify neighboring characters with the same label.

The architecture consists of a character level embedding layer that is connected to Bidirectional Recurrent Neural Network (BiRNN). Each unit of BiRNN is the Gated Recurrent Unit (Cho et al., 2014). The model outputs a probability distribution over all language tags. In order to determine the language tag of a character, we take the index of tag in the output layer with the maximum value. The architecture is illustrated in Figure 2.2.

To prevent overfitting, we use dropout 0.5 (Srivastava et al., 2014) during model training on the character embeddings. The key idea is to drop connections randomly. It prevents neurons from co-adapting too much, i.e. starting to depend on outputs of other neurons too much, which is a typical symptom of overfitting to training data.

We have collected, and processed training data containing 131 languages, which is more than other tools can recognize. We also added support to identify non-linguistic structures such as HTML codes, which are often present in noisy crawled data. For the final training set, we mixed all sources for a given language at the line level and removed the end of lines. Thus we artificially created a large code-switching training dataset.

2.2.1 Monolingual Language Identification

Most of the related research is focused on monolingual language identification, i.e. recognizing the single language of an input document. We compare our approach in this setting with several other algorithms on the dataset presented by Baldwin and Lui (2010). The dataset consists of 3 different testsets, each containing a different number of languages, styles, and document lengths collected from different sources.

Table 2.1 summarizes the accuracy of several most popular algorithms on the three testsets (Baldwin and Lui, 2010). For some algorithms, we report values as presented by Lui and Baldwin (2012) without re-running.

Despite the considerably higher number of languages covered, our model performs reasonably close to the competitors on EuroGov (testset of 10 lan-

Table 2.2: Results on our testset for short texts. The first column shows an accuracy over all 131 languages. The second column shows an accuracy over languages that all systems have in common.

System	All languages	Common languages
Langid.py	.567	.912
CLD2	.545	.891
Our model	**.950**	**.955**

guages) and Wikipedia (testset of 67 languages). We reached the best score on the TCL (testset of 60 languages).

We compare our method with two top language recognizers, Langid.py Lui and Baldwin (2012) and CLD2.[2] We train our model on more languages, and we do not restrict it to only the languages included in the testset. We did not restrict LanideNN to recognize only a subset of languages. Thus we may be losing by recognizing detailed dialect labels. Furthermore, our approach evaluates the examples on a short span of 200 characters at a time. The final prediction is based on the average predictions across the document. A different strategy of breaking the input could improve our results.

2.2.2 Short-Text Language Identification

In order to demonstrate the ability of our method to identify the language of short texts such as tweets, search queries or user messages, we wanted to use an existing corpus, such as the one released by Twitter.[3] Unfortunately, the corpus contains only references to the actual tweets, and most of them are no longer available. We thus have to rely on our testset, as described in Kocmi and Bojar (2017b), where the average line length of example is 142.3 characters.

Results on short texts are reported in Table 2.2. The two other systems, Langid.py and CLD2, are trained on texts unrelated to our collection of data and cover fewer languages. It is therefore not surprising that they perform much worse when averaged over all languages.

For a fairer comparison, we also report accuracy on a restricted version of the testset that included only languages supported by all three tested tools. Both our competitors are meant to be generally applicable, so they should (and do) perform quite well. Our system nevertheless outperforms them, reaching the accuracy of 95.5. Arguably, we can be benefiting from having trained on different texts and different distribution but the same sources as this testset.

2.2.3 Multilingual Language Identification

In multilingual language identification, systems are expected to report the set of languages used in each input document. The evaluation criterion is thus macro-

[2] https://github.com/CLD2Owners/cld2

[3] http://blog.twitter.com/2015/evaluating-language-identification-performance

Table 2.3: Results of multilingual language identification. All models uses the same training set, either ALTW2010 or WikipediaMulti. The * identifies results as reported by Lui et al. (2014).

	ALTW2010		WikipediaMulti	
System	F_M	$\mathbf{F}_\mu$	F_M	$\mathbf{F}_\mu$
* Baldwin and Lui (2010)	.464	.829	-	-
* ALTW2010 winner	.699	.932	-	-
* SEGLANG	**.784**	.905	.875	.861
* LINGUINI	.513	.700	.802	.805
* Lui et al. (2014)	.748	.933	.961	.959
Lui et al. (2014)	.724	.931	.961	.963
Our model	.779	**.965**	**.966**	**.964**

(M) or micro- (μ) averaged precision (P), recall (R) or F-measure (F).[4] The main criterion of the ALTW2010 shared task (Baldwin and Lui, 2010) was to maximize the micro-averaged F-score ($\mathbf{F}_\mu$).

To interpret the character-level predictions by our model for multilingual identification, we used the ALTW2010 development data to set the threshold empirically: if a language is predicted for more than 3 % of characters in the document, we consider the language as one of the document's languages.

We evaluate our model on two existing testsets for multilingual identification, ALTW2010 shared task, and WikipediaMulti. Both testsets come with training data. Thus we retrain our model to test its in-domain performance.

ALTW2010 shared task (Baldwin and Lui, 2010) provided 10000 bilingual documents divided as follows: 8000 training, 1000 development, and 1000 test documents. The task is to recognize which two languages are present in the document.

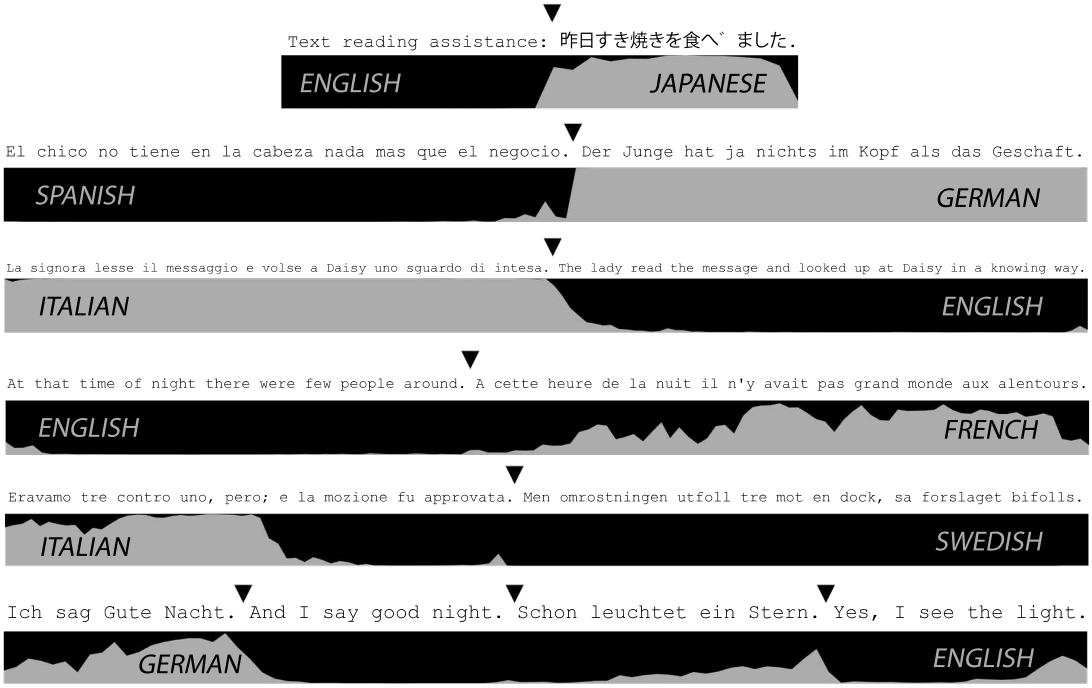
WikipediaMulti (Lui et al., 2014) is a dataset of artificially prepared multilingual documents, mixed from monolingual Wikipedia articles from 44 languages. Each of the artificial documents contains texts in $1 \leq k \leq 5$ randomly selected languages. The average document length is 5500 bytes. The training set consists of 5000 monolingual documents, the development set consists of 5000 multilingual documents, and testset consists of 1000 documents for each value of k.

The results are in Table 2.3. For algorithms SEGLANG and LINGUINI, we only reproduce the results reported by Lui et al. (2014). We use the system by Lui et al. (2014) as a proxy for the comparison: we retrain their system and obtain results similar to those reported by the original authors. The differences are probably due to the Gibbs sampling used in their approach.

We see that our model outperforms all other models in the task's main criterion $\mathbf{F}_\mu$. More details and results are in our paper Kocmi and Bojar (2017b).

[4]Note that for comparability with results reported in other works, macro-averaged F-score is calculated as average over individual F-scores instead of the harmonic mean of P_M and R_M.

Figure 2.3: Illustration of text partitioning. The black triangles indicate true boundaries of languages. The black part shows probability of detecting the language labeled in gray color, and the gray part shows complement for the second language since in this setup we restricted our model to use only the two languages in question. The misclassification of Italian and German as English in the last two examples may reflect increased noise in our English training data.



2.2.4 Code Switching Analysis

Lastly, we illustrate the ability of our model to recognize borders between languages, even on short sentences. Figure 2.3 presents the behavior of our model on text with mixed languages. The graph represents the probability of the model to recognize the first language in one color and (1–probability) for the second languages in the other color.

We have selected very short (50–130 characters) and challenging segments where the languages mostly share the same script. Finding the boundary between languages written in different scripts is not difficult, as illustrated by the first example. Moreover, it can recognize borders even on other examples, however failing on others. The more robust examination would be needed to evaluate this performance. Thus we leave it just as an illustration.

2.2.5 Conclusion

We trained language identification tool with a focus on identifying even short segments such as sentences as they are usually the smallest units gathered for MT. Furthermore, we focused on the multilingual setting. We were one of the first to tackle the problem with neural networks. We reached the best result on one of the three testset in monolingual language recognition. We outperformed

other tools in the task of short segment identification as well as multilingual language identification.

We collected a dataset and trained our model to recognize considerably more languages than other state-of-the-art tools. We have developed a language identification algorithm based on bidirectional recurrent neural networks and made it publicly available.[5]

We use this tool for analysis of transfer learning behavior in Section 5.1.1.

2.3 Training Data

This section summarizes all the datasets we use for training, development, and test evaluation throughout the whole thesis. We first introduce our high-resource Czech–English parallel corpora in Section 2.3.1, a core dataset used in most parts of our work. Then we describe the remaining datasets used in this thesis.

2.3.1 Czech–English Parallel Corpus

For our work, we have contributed to the development of a high-resource parallel corpus of Czech–English. The size of the parallel corpus is essential for training a neural machine translation model, where more data from various domains generally leads to a better general translation. Therefore, we have extended the original corpus (Bojar et al., 2012) by collecting over four times more data from more domains. The corpus contains 62.5 million sentences from various sources: Subtitles, EU Legislation, Fiction, Parallel Web Pages, Technical Documentation, Medical, PDFs from Web, News, Project Navajo, and Tweets.

CzEng 1.6 dataset is important for our work for several reasons:

- Czech language has rich morphology, which makes machine translation hard (Bojar, 2015).
- It allows Czech–English to serve as a high-resource language pair with 62.5 million data.
- It contains data from various domains.
- We understand fluently both languages, which is useful for the manual error analysis.

After the release, we have noticed that CzEng 1.6 contains a considerable number of sentence pairs with English sentences in the Czech side of the corpus or vice versa. We identified the wrong sentence pairs by automatic language identification. Initially, we wanted to use our LannideNN (see Section 2.2); however, the implementation is considerably slower than other available tools. Thus we decided to use Langid.py (Lui and Baldwin, 2012) instead to automatically check the languages.

Automatic language identification has better results for longer segments. Fortunately, the CzEng contains information about the original paragraph. We

[5] https://github.com/tomkocmi/LanideNN

Table 2.4: Datasets sizes overview. Word counts are from the original corpora, tokenizing only at whitespace and preserving the case.

		Words	
Language pair	Sentence pairs	First language	Second language
Odia–EN	27k	604k	706k
Gujarati–EN	173k	1.4M	1.4M
Estonian–EN	0.8M	14M	20M
Basque–EN	0.9M	5M	7M
Finnish–EN	2.8M	44M	64M
German–EN	3.5M	73M	77M
Slovak–EN	4.3M	82M	95M
Russian–EN	12.6M	297M	321M
French–EN	34.3M	1044M	912M
Czech–EN	40.1M	491M	563M
Arabic–Russian	10.2M	243M	252M
French–Russian	10.0M	295M	238M
Spanish–French	10.0M	297M	288M
Spanish–Russian	10.0M	300M	235M

checked the language of each of the paragraphs separately and removed all paragraphs that were identified as a different language than it should be. We removed 4M sentence pairs. This way, we filtered mostly the noisy sentences and avoided removing a large part of clean ones. Therefore the corpus still contains some noisy sentences.

We published the updated version as CzEng 1.7.[6]

Furthermore, for our experiments with transfer learning, we filtered the CzEng even more by removing short sentence pairs of 3 or fewer words, because these sentences are mostly fragments from subtitles. We also removed the longest sentences with more than 75 words as they were often only lists of items or countries[7]. The final size of the corpus we use in our work is 40.1M sentence pairs.

2.3.2 Other Datasets

We experiment with various languages across this thesis to show the generality of proposed methods. For various experiments, we select a representative subset of languages having various sizes of corpora, relatedness, translation performance, and writing scripts. Furthermore, the choice of languages was influenced by various shared tasks, where we participated (see Kocmi et al., 2018b,c; Kocmi and Bojar, 2019b).

[6] http://ufal.mff.cuni.cz/czeng/czeng17

[7] Only 0.5M parallel sentences has been removed due to having the length longer than 75 words.

Table 2.5: Language family, branch, number of speakers, and the writing script according to Simons (2018).

Language	Lang. family	Lang. branch	Speakers	Script
English	Indo-European	Germanic	1132M	Latin
German	Indo-European	Germanic	132M	Latin
French	Indo-European	Romance	280M	Latin
Spanish	Indo-European	Romance	534M	Latin
Czech	Indo-European	Slavic	11M	Latin
Russian	Indo-European	Slavic	258M	Cyrillic
Slovak	Indo-European	Slavic	5M	Latin
Gujarati	Indo-European	Indic	61M	Brahmic
Odia	Indo-European	Indic	34M	Brahmic
Arabic	Afro-Asiatic	Semitic	274M	Arabic
Estonian	Uralic	Finnic	1M	Latin
Finnish	Uralic	Finnic	5M	Latin
Basque	Basque	Basque	1M	Latin

The main criterion is the size of training corpora. We compare low-resource and high-resource language pairs spanning several orders of magnitude of training data sizes. The smallest dataset is the Odia–English with the size of 27k sentence pairs, and the biggest is the Czech–English with 40.1 million sentences. The sizes of the training datasets are in Table 2.4.

Although the Basque–English has a comparable number of sentence pairs as Estonian–English, it has only a third of the total number of words. This is due to many segments not containing a complete sentence.

The second criterion behind the selection of languages is to include language pairs reaching various levels of translation quality. This is indicated by automatic scores of the baseline setups ranging from 3.54 BLEU (English→Odia) to 36.72 BLEU (English→German),[8] see Table 4.1 on page 48.

The third criterion is language relatedness. In particular, Estonian and Finnish (paired with English) are linguistically related. Another pair of languages is Czech and Slovak, which are closely related languages with more parallel sentences.

The fourth criterion is the writing script because our methods do not need transliteration as it was a case of previous approaches. Additionally to Latin, we use languages written in Cyrillic, Brahmic, and Arabic. We present language type, number of speakers, and the writing script in Table 2.5.

[8]The systems submitted to WMT 2018 for English→German translation have better performance than our baseline because we decided not to use Commoncrawl. Thus we made German–English parallel corpus artificially less resourceful.

Table 2.6: Corpora used for each language pair in training set, development set, and the test set. The names specify the corpora from WMT News Task data except of languages from various papers.

Language pair	Trainset	Devset	Testset
English–Basque	IWSLT 2018	IWSLT dev 2018	IWSLT 2018
English–Estonian	Europarl, Rapid	WMT dev 2018	WMT 2018
English–Finnish	Europarl, Paracrawl, Rapid	WMT 2015	WMT 2018
English–German	Europarl, News commentary, Rapid	WMT 2017	WMT 2018
English–Odia	Parida et al. (2018)	Parida et al. (2018)	Parida et al. (2018)
English–Russian	News Commentary, Yandex, and UN Corpus	WMT 2012	WMT 2018
English–Slovak	Galušcáková and Bojar (2012)	WMT 2011	WMT 2011
English–French	Commoncrawl, Europarl, Giga FREN, News commentary, UN corpus	WMT 2013	WMT dis. 2015
English–Gujarati	Bible, Dictionary, Govincrawl, Software, Wiki texts, and Wiki titles	WMT dev 2019	WMT 2019

For most of the language pairs, we use training data from WMT (Bojar et al., 2018).[9] We use the training data without any preprocessing, not even tokenization.[10]

We use most of the training, development, and testsets from WMT.[11] The complete list of corpora is in Table 2.6.

For Basque–English, we use all the available data allowed by the organizers of IWSLT 2018 (Niehues et al., 2018). In addition to the resources suggested by the organizers, we used the allowed data from OPUS and WMT, specifically, corpora PaCo2 Basque–English (San Vicente et al., 2012) and QTLeap Batches 1-3 from WMT IT Translation.[12]

Our Slovak–English experiments use the corpus from Galušcáková and Bojar (2012), detokenized by Moses.[13]

The language pairs Arabic–Russian, French–Russian, Spanish–French, and Spanish–Russian, use UN corpus (Ziemski et al., 2016), which provides over 10 million multi-parallel sentences in 6 languages.

NMT suffers when the training data is not clean (Koehn et al., 2018). Based on our previous experiments, we exclude the noisiest corpus, i.e. web crawled ParaCrawl or Commoncrawl. Furthermore, language pairs with training sentences shorter than four words or longer than 75 words on either the source or the target side are removed to allow for a speedup of Transformer by capping the maximal length and allowing a bigger batch size. The reduction of training data is small, and it does not change the performance of the translation model.

In contrast, for French–English we keep all WMT 2018 corpora and perform a quick cleaning using language detection by Langid.py (Lui and Baldwin, 2012). We drop all sentences that are not recognized as the correct language. This cleaning removes 6.5M sentence pairs from the French–English training corpus.

[9] http://www.statmt.org/wmt18/

[10] While the recommended best practice in past WMT evaluations was to use Moses tokenizer, it is not recommended anymore for Tensor2Tensor with own build-in tokenizer.

[11] http://www.statmt.org/wmt18/

[12] http://www.statmt.org/wmt16/it-translation-task.html

[13] https://github.com/moses-smt/mosesdecoder

We often abbreviate English as EN to visibly differentiate it from the second language.

2.4 Machine Translation Evaluation

In order to evaluate how successful the machine is in translating, we need to define what is considered a good translation. It inherently leads to defining when two texts, in a different language, constitute the equivalent meaning. It is a difficult task, and we would need to delve into complex theoretical questions, which is out of the scope of this thesis. Thus MT researchers usually evaluate MT translations by comparing it to the expert human translations.

MT evaluation is usually focusing on two main aspects called fluency and adequacy. Whenever the system produces syntactically well-formed sentences (i.e. high fluency) and does not change the semantics, the meaning of the source sentence (i.e. high adequacy), it is considered as a good translation (Hovy et al., 2002). Various ways of measuring the fluency have been proposed, and new metrics are annually evaluated in the WMT shared task (Ma et al., 2018). As for the adequacy, it is more complicated since multiple correct translations are possible, and therefore, it is mostly evaluated by conducting the manual evaluation by humans, which is time-consuming and costly.

Bojar et al. (2013) created a method for the generation of millions of possible references that could solve the problem with having only a limited number of references (usually only one). However, their testset is restricted only to 50 prototype sentences.

2.4.1 Manual Evaluation

Manual evaluation campaigns are run each year at Workshop on Statistical Machine Translation (WMT) to assess translation quality of both academic and commercial systems. This evaluation is considered a benchmark for identifying state-of-the-art systems of a given year.

Manual evaluation utilizes human ability to judge what is a good translation without a rigorous definition. Throughout the years, the WMT manual evaluation has changed several times based on findings from previous years. For example, comparing multiple systems together, binary yes/no decision about the translation, or by directly rating one translation at a time as is the case of last three years (Graham et al., 2017). The main idea across the approaches remains similar: Crowd-sourced judges are asked to rank presented outputs from various systems based on their intuition.

However, due to the high demand for cost and time, we are not using the manual score in this thesis. Instead, we rely on automatic metrics that try to replicate human behavior as closely as possible.

2.4.2 Automatic Metrics

Automatic metrics for MT evaluations are often based on the estimation of similarity between the system output, and a human-produced reference translation. The most often used metric is the BLEU score (Papineni et al., 2002). It is based

on comparing n-grams of sentence units, typically words, between the system output and one or more reference sentences. It is computed as the geometric mean of n-gram precisions for $n = 1...N$ with penalization for short translations by brevity penalty, according to the following formula:

$$BP = \begin{cases} 1 & \text{if } L_{sys} > L_{ref} \\ e^{\left(1-L_{ref}/L_{sys}\right)} & \text{if } L_{sys} \leq L_{ref} \end{cases} \tag{2.1}$$

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \cdot \log p_n\right) \tag{2.2}$$

where w_n is a positive weight summing to one, usually $\frac{1}{N}$. L_{ref} denotes the length of the reference text that is closest in length to the system output, L_{sys} is the length of the system output. The standard value of N for BLEU is 4, different n-gram lengths are rarely used. The n-gram precision p_n is computed by dividing the number of matching n-gram in the system output by the number of considered n-grams. The number of n-gram matches are clipped to the frequency in the reference when n-grams occur multiple times. BLEU is a document-level metric. Thus the counts of confirmed n-grams are collected for all sentences in the document (or testset) and then the geometric mean of n-gram precision is computed from the accumulated counts.

It is more informative to compare system output against several references, but it is expensive to obtain multiple references. Thus only one reference is used in most of the testsets.

There is a numerous criticism that has been observed of BLEU. For instance:

- use of a geometric mean, which makes the score 0, when there is no match at any of the n-gram levels (usually a problem of short testsets);
- gives no credit for synonyms or different inflected forms of the same word;
- does not consider the importance of various n-grams;
- it is too sensitive to tokenization.

There are several studies on the reliability of BLEU (Callison-Burch et al., 2006; Bojar et al., 2010), which inspired the development of other metrics (Lavie and Agarwal, 2007; Ma et al., 2018). Despite the criticism and other possible metrics, BLEU remains the standard metric for automatic evaluation of MT systems.

There are several implementations of BLEU that differ in various tokenization details, case-sensitivity, and other details that lead to different resulting scores. Post (2018) made a call for clarity in reporting BLEU score and implemented SacreBLEU[14] tool for more comparable results. The evaluation script automatically downloads reference testset and computes performance using various metrics. In this work, we use SacreBLEU with the same setting whenever we are reporting numerical results.[15]

We report BLEU score multiplied by 100 as is it usual in most papers instead of values on the interval 0 to 1 as originally described by Papineni et al. (2002).

[14] `https://github.com/mjpost/sacreBLEU`

[15] SacreBLEU signature is: `BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.2.1`

2.4.3 Statistical Significance

If two translation systems differ in BLEU performance, it does not necessarily mean one is significantly better than the other. In order to indicate the actual quality, Koehn (2004) proposed to use the paired bootstrap resampling method to compute the statistical significance and validate the superiority of one of the systems. The method repetitively creates testsets by drawing sentences from the original testset randomly with repetition and evaluating the automatic score. Then it computes statistical confidence overall scores comparing various MT systems.

Whenever we talk about statistical significance in this work, we tested the compared systems by paired bootstrap resampling with 1000 samples and the confidence level of 0.05. In the results, we label it with a ‡ symbol and comment it in the text. We perform significance tests, usually comparing the baseline and an examined system (if not specified otherwise in the text).

We do the pair-wise statistical testing as is customary in NMT.

3 Neural Machine Translation

In this section, we describe NMT. In Section 3.1, we explain in detail word embeddings, NMT part that is crucial for our work. We describe subword representation and how NMT handles Out-of-Vocabulary (OOV) words in Section 3.2. Then we describe the whole NMT architecture in Section 3.3. We describe the toolkit we used and detailed settings for our model in Section 3.4. Lastly, we describe how the progress of NMT training is measured and propose stopping criterion in Section 3.5.

3.1 Word Embeddings

Neural networks work in continuous space. When used for Natural Language Processing (NLP) tasks, we need to bridge the gap between the world of discrete units of words and the continuous, differentiable world of neural networks.

Originally, the first step was to use a finite vocabulary, where each word had a different index, which was then represented as a one-hot vector of the size equal to the number of items in vocabulary. Sizes of 10–90k words were used in NLP. However, the one-hot representation, a vector containing only zeros except at a single position with one, is not continuous and differentiable. Furthermore, it does not generalize such that similar words are closer together within the representation.

One way of compressing the one-hot vectors is to assign each word a specific dense vector through an NN layer can be called *lookup tables* or *word embeddings*.

Embeddings (Bengio et al., 2003) are dense vector representations of words commonly of 100-1000 dimensions. They are trained jointly with the whole network and learn word-specific features and cluster the words in the space so that similar words have vectors that are close to each other.

Mikolov et al. (2013) found that word embeddings in language model NN contain semantic and syntactic information without being trained to do so. An example of embedding clustering the space of words is in Figure 3.1.

Figure 3.1: Thirty nearest neighbors in cosine similarity for the word “woman” visualized in 2D by principal component analysis. The large color clusters were added manually for better presentation. The representation is from the encoder BPE subword embeddings of our Czech→English model. This figure shows that the 30 nearest neighbors are variants of the word “woman”. Interestingly there are two separate clusters for Czech and English words (blue and pink), which suggests that NN understands equivalence across languages. Furthermore, there are clusters dividing words for adult women and young women (green and orange). Worth of mentioning is the subword “kyně”, which is a Czech ending indicating the feminine variant of several classes of nouns, e.g. professions. It appears in the “young women” cluster probably because of the common word “přítelkyně” (girlfriend).

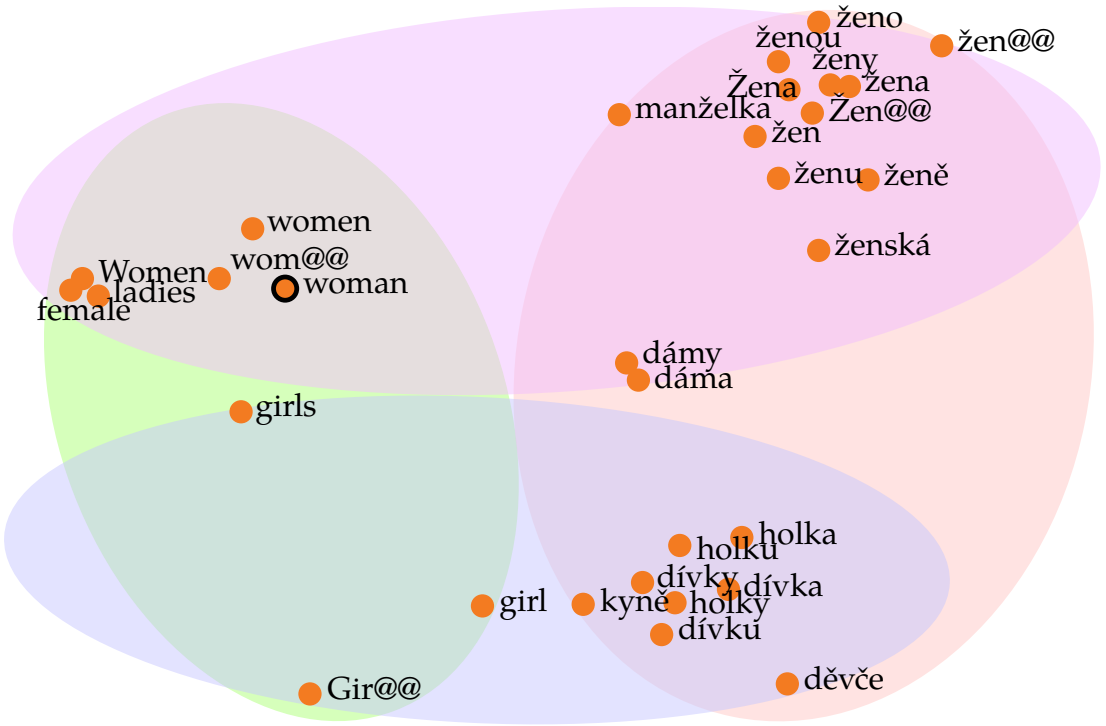


Table 3.1: Examples from Mikolov et al. (2013) testset question types, the upper part are semantic questions, the lower part is considered syntactic by Mikolov et al. (2013).

Question Type	Sample Pair
capital-countries	Athens – Greece
capital-world	Abuja – Nigeria
currency	Algeria – dinar
city-in-state	Houston – Texas
man-woman	boy – girl
adjective-to-adverb	calm – calmly
opposite	aware – unaware
comparative	bad – worse
superlative	bad – worst
present-participle	code – coding
nationality-adjective	Albania – Albanian
past-tense	dancing – danced
plural	banana – bananas
plural-verbs	decrease – decreases

Word embeddings can exhibit an interesting correspondence between lexical relations and arithmetic operations in the vector space. The most famous example is the following:

$$v(\textit{king}) - v(\textit{man}) + v(\textit{woman}) \approx v(\textit{queen})$$

In other words, adding the vectors associated with the words '*king*' and '*woman*' while subtracting '*man*' should be close to the vector associated with the word '*queen*'. We can also say that the difference vectors $v(\textit{king}) - v(\textit{queen})$ and $v(\textit{man}) - v(\textit{woman})$ are almost identical and describe the gender relationship.

Mikolov et al. (2013) noticed that such relations emerge without specific training criteria naturally from training the language model with unannotated monolingual data.

3.1.1 Lexical Relations Testset

In order to test lexical relations learned by word embeddings, Mikolov et al. (2013) proposed a testset of question pairs. Each question contains two pairs of words (x_1, x_2, y_1, y_2) and captures relations like "What is to 'Paris' (y_1) as 'Czechia' (x_2) is to 'Prague' (x_1)?", together with the expected answer 'France' (y_2). The model is evaluated by finding the word representation that has the nearest cosine similarity to the vector $vec(\text{Czechia}) - vec(\text{Prague}) + vec(\text{Paris})$. If the nearest neighbor is $vec(\text{France})$, we consider the question answered correctly.

The Mikolov et al. (2013) testset consists of 19544 questions, of which 8869 are called semantic, and 10675 are called syntactic, and further divided into 14 types, see Table 3.1.

Table 3.2: The statistics of Mikolov et al. (2013)'s and our testset. The size represents the total number of questions in the testset.

Testset	Size	Categories	Unique words
Mikolov et al. (2013) (syntactic)	10675	9	537
Our testset	8000	8	5424

After a closer examination of the dataset, we found out that it does not test what the broad terms syntactic and semantic relations suggest. Questions of only three types cover semantics: predict a city based on a country or a state, currency name from the country and the feminine variant of nouns denoting family relations. Vylomova et al. (2016) showed that many other semantic relationships could be tested, e.g. walk-run, dog-puppy, bark-dog, cook-eat, and others. For the "syntactic" relations, the testset contains mostly frequent words that often regularly form morphological variants (e.g. by adding the suffix 'ly' to change an adjective into the corresponding adverb), which decreases the generality of the testset.

We have decided to extend the morphosyntactic relations in the testset by adding a substantial number of morphological variants. We extended the syntactic questions except for nationality adjectives, which are already completely covered in the original testset.

We constructed the pairs taking inspiration in the Czech side of the CzEng corpus (see Section 2.3.1), where explicit morphological annotation allows identifying various pairs of Czech words (different grades of adjectives, words, and their negations among others). Word-aligned English words often shared the same properties. As further sources of pairs, we used various webpages usually written for learners of English. For example, for verb tense, we relied on a freely available list of English verbs and their morphological variations. We have included 100–1000 different pairs for each question set. The questions were constructed from the pairs similarly as by Mikolov et al. (2013): generating all possible pairs of relation pairs. All combinations lead to millions of questions, so we randomly down-sampled to 1000 instances per question set, to keep the testset in the same order of magnitude as in the original one. Additionally, we decided to extend the set of questions on opposites to cover not only opposites of adjectives but also of nouns and verbs.

The comparison of the testsets is provided in Table 3.2. Our testset contains a similar number of questions per category as the original. However, the number of unique words is more than ten times higher, which is useful in diagnosing rare word relations.

Our testset is publicly available. Further details can be found in our paper Kocmi and Bojar (2016).

3.1.2 SubGram Representation

The Skip-gram model (Mikolov et al., 2013) uses one-hot representation of a word in vocabulary as the input vector x. The embedding of a word then corresponds to the multiplication of the one-hot vector with the trained weight matrix (the lookup table). Therefore weights w_i of the input word i can be directly used as word embeddings E:

$$E_j = \sum_{i=1}^{|x|} x_i * w_{ij} = w_j \tag{3.1}$$

In Kocmi and Bojar (2016), we propose a substring-oriented extension of Skip-gram model that induces vector embeddings from the character-level structure of individual words. Our approach gives the NN more information about the examined word reducing the issue of data sparsity and introducing the morphological information about the word to NN.

Our approach provides the neural network with a "multi-hot" vector representing the substrings contained in the word instead of the one-hot vector representing the whole word.

We use a vocabulary of substrings, instead of words, created in the following fashion: first, we take all character bigrams, trigrams, tetragrams, and so on up to the length of the word. This way, even the word itself is represented as one of the substrings. As an indication of the beginning and the end of words, we appended the characters ˆ and $ to each word. Here we provide an example of the segmentation:

'cat' = {'ˆc', 'ca', 'at', 't$', 'ˆca', 'cat', 'at$', 'ˆcat', 'cat$', 'ˆcat$'}

Using all possible substrings would increase the size of vocabulary beyond a reasonable size. Thus we select only the most frequent substrings based on the frequency in the training data.

In order to generate the vector of substrings, we segment the word and create a multi-hot vector, where "ones" indicate word's substrings indices in the vocabulary. In other words, each word is represented as a multi-hot vector indicating which substrings appear in the word.

The word embedding is created in the same fashion as in the one-hot representation: by multiplication of the input vector with the weight matrix. We have to keep in mind that each word has a different number of substrings. Thus the embeddings need to be normalized either by sigmoid function or by averaging over the number of substrings. We decided to use the mean value as it is computationally simpler than sigmoid:

$$E_j = \frac{\sum_{i=1}^{|x|} x_i * w_{ij}}{\sum_{i=1}^{|x|} x} \tag{3.2}$$

where x is the multi-hot vector, and the summation in denominator represents the number of found substrings of the word.

Table 3.3: The accuracy (in %) of word embeddings on the word similarity testsets. The original testset (Mikolov et al., 2013) does not contain many OOV words, thus the score cannot be computed. Our testset is constructed in a similar way as the original, although it is more challenging and contains many OOV question pairs.

	Skip-gram		SubGram	
Testset	All	Only OOV	All	Only OOV
Original semantic	47.7	–	0.0	–
Original syntactic	42.5	–	42.3	–
Our testset	9.7	0.0	22.4	1.6

SubGram Results

We conclude this section with the evaluation of the SubGram on the Lexical relation testsets as described in Section 3.1.1.

Table 3.3 reports the results from our paper (Kocmi and Bojar, 2016). It compares Skip-gram (Mikolov et al., 2013) with our SubGram. We trained both with the same framework (Řehůřek and Sojka, 2010) on the same training data. By comparing both approaches on the original testset, we see that both algorithms reach overall a similar performance in the syntactic pairs. On the other hand, SubGram does not capture the tested semantic relations at all.

When comparing models on our testset (see Section 3.1.1), we see that given the same training set, SubGram significantly outperforms Skip-gram model. Furthermore, our testset contains many questions with rare words to test the capability to encode OOV by our SubGram model. The results show that our model can capture a small fraction (1.6%) of relations on the OOV part of testset compared to flat zero for Skip-gram. The performance on OOVs is expected to be lower since the model has no knowledge of syntactic exceptions and can only benefit from regularities in substrings.

To conclude, our model, compared to Skip-gram, can encode even unseen words, has a comparable or better performance on syntactic tasks and shows some performance on the OOV part of the testset. It could be useful for NLP tasks that do not produce any textual output, for example, sentiment analysis, language identification, or Part-of-Speech (POS) tagging. However, the approach is not reversible, and there is no simple way to transform embeddings back to word forms, which would be needed in word generation, such as the target side of machine translation. However, the inability to decode the embedding back to the word form and the not so high performance on OOVs were the main reasons why we decided not to test the SubGram in NMT. In the following section, we describe two approaches for solving the OOV problem that later became widely used in the NMT.

Table 3.4: Task performance with various embedding initializations. The higher the score, the better, except for the LM perplexity. The best results for random (upper part) and pretrained (lower part) embedding initializations are in bold. The * marks comparably performing settings in each category (random/pretrained).

Initialization	NMT (BLEU)	LM (Perplexity)	TAG (%)	LEM (%)
$\mathcal{N}(0, 10)$	6.93	76.95	85.2	48.4
$\mathcal{N}(0, 1)$	9.81	61.36	87.9	94.4
Only ones	10.63	62.04	90.2	95.7
* $\mathcal{N}(0, 0.1)$	11.77	56.61	90.7	95.7
* $\mathcal{N}(0, 0.01)$	11.77	56.37	**90.8**	**95.9**
* $\mathcal{N}(0, 0.001)$	**11.88**	**55.66**	90.5	**95.9**
* Only zeros	11.65	56.34	90.7	**95.9**
* He et al. (2015)	11.74	56.40	90.7	95.7
* Glorot and Bengio (2010)	11.67	55.95	**90.8**	**95.9**
* Word2Vec	12.37	**54.43**	90.9	95.7
GloVe	11.90	55.56	90.6	95.5
* Self-pretrain	**12.61**	54.56	**91.1**	**95.9**

3.1.3 Word Embedding Initialization

Initialization of weights for various NN layers is known to be critical for the final performance of the model. And bad initialization can doom the training altogether (Glorot and Bengio, 2010; Mishkin and Matas, 2016). The rest of the thesis is devoted to this topic, and here we start with the earliest part, word embeddings.

Various studies have provided valuable information on initialization of weights for various parts of the NN (Glorot and Bengio, 2010; He et al., 2015; Kumar, 2017). Up until our study Kocmi and Bojar (2017c), there has been a lack of research examining the initialization of word embeddings, which has distinctive properties compared to inner NN layers, e.g. it is the first layer of the network, and its input is a discrete one-hot vector.

Traditionally, word embeddings were initialized either randomly with uniform or normal distribution with small variance and a zero mean. Another option is to take word embeddings from the model trained on the same task's training data called "self-pretrain" or on a different task, usually language modeling, which can be trained on abundant monolingual data.

In practice, random initialization of embeddings is still more common than using pretrained embeddings, and it should be noted that pretrained embeddings are not always better than random initialization (Dhingra et al., 2017).

In our study (Kocmi and Bojar, 2017c), we investigated various random as well as pretrained initialization of embeddings to determine the best approach

on four tasks: English→Czech NMT, LM, part-of-speech tagging (denoted TAG), and lemmatization (LEM). For further details see Kocmi and Bojar (2017c)

The results are presented in Table 3.4. We found out that the embeddings are not prone to bad random initialization and do not need to be set precisely to maximize the performance as it is a case for the inner layers (Mishkin and Matas, 2016). As long as the variance is low, up until 0.1 for normal distribution, the NN trains ideally and the final performance is comparable.

The most surprising result of our work is that embeddings initialized with only zeros performed equally well as random initializations. If all the weights in the NN are initialized with zero, the derivative with respect to loss function is the same for every weight in each layer. Consequently, the network would train to be symmetric at each layer and would not be better than the linear model. However, it is not the case when only embeddings start with zeros, and the rest of the network is initialized randomly with various distributions.

Our results on the transferring of pretrained word embeddings support previous findings that these embeddings improve the performance over random initialization (Kim, 2014; Lample et al., 2016). However, the pretrained embeddings have a disadvantage that the model to which they are transferred has to have the same vocabulary and embeddings dimensions.

To our knowledge, our work was the first that compared various initialization techniques of the embeddings on multiple tasks. However, our work has been done on word-level NN. Since then, the research focus shifted the standard approach from the word-level translation to subword-level translation (Sennrich et al., 2016b).

Transferring of pretrained word embeddings have proven to be invaluable for improving performance of natural language tasks that often suffer from lack of training data (Kim, 2014; Lample et al., 2016), thanks to the utilization of unsupervised pretraining on a large quantity of monolingual data. However, it is less common in NMT to utilize the pretrained embeddings, mostly because the number of available parallel sentences for various language pairs tends to be several times larger than available annotated data for other NLP tasks such as Penn Treebank (Marcus et al., 1993) for parsing. However, Qi et al. (2018) showed that the use of pretrained embeddings dramatically improves the performance of the model in the low-resource NMT scenario. Especially for the extremely low-resource languages with less than 20k parallel sentences, pretrained embeddings help to improve the performance of up to 10 BLEU points.

Recently, pretrained embeddings have been used in unsupervised NMT, where the goal is to train an MT system without any parallel sentences (Artetxe et al., 2018; Lample et al., 2018) with only the monolingual data in both languages. They used the capability of embeddings to represent the meaning of words (see Section 3.1), and with the use of linear transformation, they mapped the embedding spaces of the two languages into the same vector space. Then they use this mapping to roughly translate the monolingual sentences word-by-word, and in following steps, they iteratively train the NMT system.

In recent years the research focus has shifted from word-level NMT to subword-level NMT, which made the use of pretrained embeddings in NMT obsolete. Although, several notable improvements in language modeling (De-

vlin et al., 2019; Yang et al., 2019) have appeared recently, which can lead to restoring a research focus in pretrained embeddings.

3.2 Subword Representation

Traditionally NMT systems relied on a vocabulary to store all words used in the translation. The capacity of this vocabulary was typically 10–90k words. However, this is not enough to cover all words in a language. That is why the first NMT systems used a special OOV symbol as a replacement for remaining rare words.

The Out-of-Vocabulary (OOV) words are a substantial problem especially for languages with inflection, agglutination or compounding, where many variants of a frequent word become rare. For example consider German compound word 'Abwasser|behandlungs|anlange' for 'sewage water treatment plant' or Czech 'velko|výroba' for 'mass-production', for which a segmented representation is more informative than one vector for the whole word.

Increasing the size of the vocabulary in order to reduce the number of OOV words proportionally increases the training complexity as well as decoding complexity. Moreover, the NMT systems will not be able to learn good encoding for uncommon words or word-forms as they are seen only a few times within the training corpus or not at all.

To overcome the large vocabulary problem and avoid the OOV problem, translation models need mechanisms that go below the word level. There are two possible solutions. Either by including more information into the representation of words or splitting uncommon words and translating on the level of subword units.

The former approach tries to encode additional information about the word structure or linguistic classes into the representation of a word. Tamchyna et al. (2017) removed the inflection by morphologically annotating training sentences and let the NMT to translate only the lemmas and assigned morphological tags. Luong and Manning (2016) proposed to use a hybrid approach where first we get the word embeddings from characters followed by standard NMT on the computed embeddings. In Kocmi and Bojar (2016), we proposed to include the substring structure of a word into the word embedding (see Section 3.1.2).

The latter approach breaks uncommon words into subword units that are handled by the NN as standalone tokens. The trivial approach is to break the sentence into individual characters, but it needs much longer training times as the number of tokens per training example is several times higher than the number of words, and it creates a problem with long-range dependencies in characters making the character-level translation sub-optimal (Tiedemann, 2009; Ling et al., 2015). Thus we need to split the words into the least number of subwords but avoid bloating the size of the subword vocabulary.

In recent years, several segmentation algorithms have been proposed, however, only the byte pair encoding (Sennrich et al., 2016b) and wordpieces (Wu et al., 2016) became widely used. We describe both of the methods in the next sections. For the completeness, Kudo and Richardson (2018) recently developed a segmentation method called SentencePiece, which shows promising improvements in performance over them.

Figure 3.2: BPE merges learned from a vocabulary {'old','older','wider'}.

r	</w>	→	r</w>
o	l	→	ol
e	r</w>	→	er</w>
d	er</w>	→	der</w>
w	i	→	wi

3.2.1 Byte Pair Encoding

Using a word-based vocabulary in NMT leads to problems with OOV. Sennrich et al. (2016b) tackled this problem by segmenting the words into more frequent subword tokens with the use of byte pair encoding (Gage, 1994).

BPE is a simple data compression algorithm, which iteratively merges the most frequent pairs of consecutive characters or character sequences. A table of the merges, together with the vocabulary, is then required to segment a given input text.

The table of merges is generated in the following way. First, all characters from the training data are added into the vocabulary plus a special symbol for the word ending '$\langle/w\rangle$', which is used to restore original segmentation after the translation. Then we add the ending symbol to all words in the training set and separate them to individual characters. We iteratively find the most frequent symbol pairs and replace them with a new single symbol representing their concatenation. Each merge thus produces a new symbol that represents a character n-gram. We continue until we have the same number of initial characters and merges as is our desired size of the vocabulary. By this process, frequent words become directly included in the vocabulary. A toy example is in Figure 3.2.

The merges are applied in advance on the training corpus by merging characters based on learned merges. The BPE segments the words into subword tokens, which can be used by NMT without any need for architecture modification. In other words, the NMT model handles subwords as regular words.

In practice, the symbol for the end of the word is not produced during segmentation. Instead, a '@@' is added to all subword tokens that end in the middle of a word. For example the word 'older' would be segmented into 'ol@@ der', see Figure 3.2.

Sennrich et al. (2016b) also showed that using joint merges, generated from concatenated trainsets for both the source and the target language, is beneficial for the overall performance of NMT. This improved consistency between the source and target segmentation is especially useful for the encoding of named entities, which helps NMT in learning the mapping between subword units.

BPE implementation has several disadvantages. It cannot address well languages that do not use a space as a separator between words, for example, Chinese. It fails when encoding characters that are not contained in the vocabulary, for example, foreign words written in a different alphabet. Lastly, BPE algorithm relies on a tokenizer. Without its use, the punctuation attached

directly to words would have different word segmentation than when separated. The wordpiece method (Wu et al., 2016) solves all these problems. We describe it in the next section.

3.2.2 Wordpieces

Wordpiece is another word segmentation algorithm. It is similar to BPE and is based on an algorithm developed by Schuster and Nakajima (2012). Wu et al. (2016) adopted the algorithm for NMT purposes. We describe the algorithm in comparison to BPE.

The wordpiece segmentation differs mainly by using language model likelihood instead of highest frequency pair during the selection of candidates for new vocabulary units. Secondly, it does not employ any tokenization leaving it for the wordpiece algorithm to learn.

The algorithm works by starting with the vocabulary containing units of characters and building a language model on the segmented training data. Then it adds a combination of two units from the current vocabulary by selecting the pair of units that increases the likelihood on the training data the most, continuing until the vocabulary contains the predefined number of subword units.

The iterative process would be computationally expensive if done by brute-force. Therefore the algorithm uses several improvements, e.g. adding several new units at once per step or testing only pairs that have a high chance to be good candidates.

The segmentation works in a greedy way when applying. It finds the longest unit in the vocabulary from the beginning of the sentence, separates it and continues with the rest of the sentence. This way, it does not need to remember the ordering of merges; it remembers just the vocabulary. This makes it simpler than BPE.

The Tensor2tensor (Vaswani et al., 2018) framework slightly improves the wordpiece algorithm by byte-encoding OOV characters, which makes any Unicode character encodable. It uses an underscore instead of '$\langle/w\rangle$' as an indication of the word endings.

Furthermore, the implementation by Vaswani et al. (2018) optimizes the generation by counting frequencies for only a small part of the corpus. We extended it and created vocabularies in this thesis from the first twenty million sentences. Additionally, (Vaswani et al., 2018) introduce a 1%[1] tolerance for the final size of the vocabulary. Therefore instead of having 32000 subwords,[2] the vocabulary has between 31680 and 32320 items, see `https://github.com/tensorflow/tensor2tensor/blob/v1.8.0/tensor2tensor/data_generators/text_encoder.py#L723`.

Whenever we talk about the wordpiece segmentation in this work, we mean the T2T implementation described in this section.

[1] The implementation in Tensor2tensor (T2T) tries to create vocabulary several times, and if it fails to create a vocabulary within this tolerance, it uses the generated vocabulary with the closest size.

[2] We use exactly 32000 as a vocabulary size instead of $2^{15} = 32768$.

3.3 Neural Machine Translation Architectures

There were early attempts to use neural networks in machine translation (Waibel et al., 1991; Forcada and Ñeco, 1997). However, the rise of neural networks in MT came two decades later, when the hardware resources became able to handle large models with millions of parameters.

A shift in paradigm in MT happened when NMT end-to-end model scored higher that previous PBMT models (Jean et al., 2015; Sennrich et al., 2016d).

One of the first end-to-end NMT systems was Sutskever et al. (2014). They used Long Short-Term Memory cells (LSTM) Recurrent Neural Network (RNN) model that processes one word at a time until it reads the whole input sentence. Then a special symbol "<start>" is provided and the network produces the first word based on its inner state and the previous word. This generated word is then fed into the network, and the second word is generated. The process continues until the model generates the "<end>" symbol.

The main disadvantage of the work of Sutskever et al. (2014) is that the network has to fit the whole sentence into a vector of 300–1000 dimension before it starts generating the output. Therefore Bahdanau et al. (2014) proposed the so-called attention mechanism. The attention mechanism gives the network the ability to reconsider all input words at any stage and use this information when generating a new word.

Gehring et al. (2017) redesigned the previous architecture with Convolutional Neural Network (CNN), which handles all input words together, therefore making the training and inference process faster.

Vaswani et al. (2017) completely redesigned the NMT architecture and introduced Transformer model, which uses feed-forward layers in contrast to previous architectures that use RNN or CNN structures. We describe this architecture in detail in the next section.

3.3.1 Transformer Model

In our work, we use the Transformer architecture (Vaswani et al., 2017). Transformer architecture consists of an encoder and decoder, similarly as the previous approaches. The encoder takes the input sentence and maps it into a high-dimensional state space. Its output is then fed into the decoder, which produces output sentence. However, instead of going one word at a time from left to right of a sentence, encoder sees the entire input sequence at once. This makes it faster in terms of training and inference speed in comparison to previous neural architectures because it allows better usage of parallelism. The decoder remains "autoregressive", i.e. always producing the output symbol with the knowledge of the previously produced output symbol. Non-autoregressive models (Libovický and Helcl, 2018) are still an open research question.

Transformer Attention

The idea of attention mechanism (Bahdanau et al., 2014) is to look at the input sequence and decide which words are important for the generation of a particular output word. The novel idea of self-attention is to extend the mechanism to the processing of input sequences and output sentences as well. In other words, it

helps the model to understand the word it is currently processing with the use of relevant words from its context.

For example, when processing the sentence "The kitten crawled over the room because it was hungry.", NMT needs to know the antecedent of the word "it". The self-attention mechanism solves this problem by incorporating the information into the representation of the word "it" at deeper layers of the encoder.

In general form, the Transformer attention function uses three vectors: queries (Q), keys (K) and values (V). The output is a weighted sum of values, where weights are computed from queries and keys. The attention is defined as follows:

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

where d_k is the square root of the dimension of the key vectors, which is normalization necessary to stabilize gradients.

The intuition behind the attention is that we get a distribution over the whole sequence using the dot product of queries (which are a hidden state of all positions in the sequence) and keys followed by softmax. This distribution is then used to weight the values (which encode a hidden state similarly like queries). It results in a vector, where relevant words or their features are stressed.

The attention is used separately in encoder and decoder as a self-attention where all queries, keys, and values come from the previous layer. It is also used in encoder-decoder attention, where queries and keys come from encoder and values from the decoder.

Multi-Head Attention

Having only one attention, NMT would focus solely on some positions in the previous layer, leaving other relevant words ignored or conflating mutually irrelevant aspects into one overused attention. Transformer model solves this by using several heads within each layer, each with its own linear transformation, which leads to the concurrent observation of different parts of the input.

Formally, the multi-head attention is defined as follows:

$$\text{MultiHead}(Q, K, V) = \text{Concatenate}(\text{head}_1, ..., \text{head}_h)W^O$$
$$\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

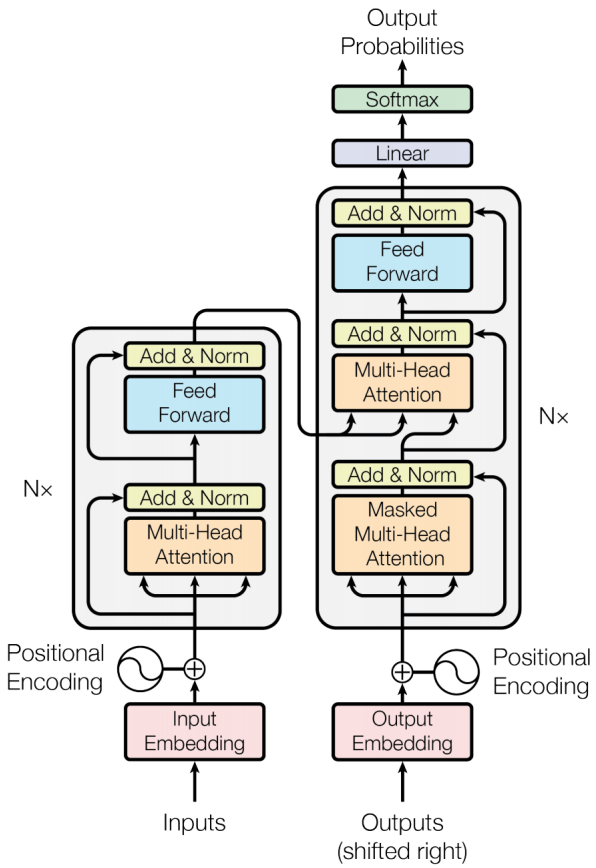
where the projection matrices $W^{Q/K/V}$ are trainable matrices different for each attention head and h is the number of heads, sixteen in the "Transformer-big" model. The concatenation in multi-head attention is then linearly projected by a matrix W^O.

Positional Encoding

With the use of attention mechanism, one thing is missing from the model. It is the information about the position of each word. This is solved by adding a special positional encoding to all input words, which helps NMT to identify the word order.

The absolute position encoding of a word pos is defined as:

Figure 3.3: Transformer architecture. The image is taken from Vaswani et al. (2017).



$$PE_{(\text{pos},2i)} = \sin(\text{pos}/10000^{2i/d_{model}})$$

$$PE_{(\text{pos},2i+1)} = \cos(\text{pos}/10000^{2i/d_{model}})$$

where i is the dimension in the positional encoding PE, in other words, each dimension of PE corresponds to a sigmoid.

In Section 3.1, we explained how discrete words are mapped to embedding vectors. The positional encoding is added to the word embeddings and used as the input to the first layer of Transformer.

Complete Architecture

The complete architecture is illustrated in Figure 3.3. Except parts mentioned above, there is a residual connection after each multi-head attention, which sums input of multi-head attention with its output followed by layer normalization (Ba et al., 2016) (it is labeled as "add & norm" in Figure 3.3). The model stacks several layers of multi-head attention on top of each other, with position-wise feed-forward networks. In the original model, six layers are used.

The output of the decoder is finally modified by linear transformation followed by the softmax function that produces probabilities of words over the model vocabulary.

For further reading, see the original paper (Vaswani et al., 2017) or various blog posts describing the model.[3]

3.4 Neural Machine Translation Model Setting

In our earlier works on NMT, we used the Neural Monkey toolkit and the RNN model with an attention mechanism. The Neural Monkey has been developed mainly for quick prototyping, see further details in Helcl et al. (2018).

Vaswani et al. (2017) published their work with the Transformer model, which showed significant gains in the training time and performance. Furthermore, they published the T2T framework (Vaswani et al., 2018). For the majority of this thesis, we use the T2T framework.

During our research, T2T has been rapidly developed and improved. Therefore, we changed to the newer version twice throughout our research. This means that the results should not be directly compared as any change in the network architecture or training regime may lead to large changes in the performance.

In this thesis, each section uses one of three different settings that we explain in this section. Each of the settings has been used in our papers, and results are comparable within each setting. However, the results for same language pair should not be compared across settings. We use the following setups:

- **T2T**$_4$ – this setup uses T2T in version 1.4.2. This setting is used in the "warm-start" experiments in Kocmi and Bojar (2018); Kocmi et al. (2018b,c,a).
- **T2T**$_8$ – this setup uses T2T in version 1.8.0 and concerns around the "cold-start" experiments in Kocmi and Bojar (2019a).
- **T2T**$_{11}$ – our latest setup uses T2T in version 1.11.0 and is involved in Kocmi and Bojar (2019b). Furthermore, we use this setup for experiments that are not yet published.

We use a model based on the "Big single GPU Transformer" setup as defined by Vaswani et al. (2017) with a few modifications. We set maximal length of a sentence to 100 wordpieces. We set a length normalization penalty to 1. The individual versions use additional modifications:

- **T2T**$_4$ uses Adam optimizer (Kingma and Ba, 2015) with 32000 linear warm-up steps, and batch size of 2300 tokens.
- **T2T**$_8$ uses Adafactor optimizer (Shazeer and Stern, 2018) with 8000 linear warm-up steps, batch size of 2900 tokens, and disabled layer dropout.
- **T2T**$_{11}$ uses Adafactor optimizer with 16000 linear warm-up steps, batch size of 4500 tokens, and disabled "layer dropout".

For details about individual parameters, see Vaswani et al. (2018).

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

3.5 Measuring Training Progress

The progress of NN training can be measured in various ways. We can measure, for example, the wall-clock time passed, the number of processed training examples or the financial cost of the training. Each method is more relevant for various applications.

The most common approach is to report the number of processed training steps. It can be reported either by the number of individual seen examples, number of training epochs or as a number of batches. The individual examples are preferred in tasks, where each example has the same informative value. In MT, one training example is usually a sentence pair, which can be of varying length. Reporting the number of epochs does not rely on the ordering of sentences in corpus because it is reported after each complete pass over the training corpus. Lastly, we can report the number of batches, also called training steps, which specifies the number of updates (error backpropagation) through the network.

The batch size can be either fixed to a given number of sentence pairs (e.g. in Neural Monkey Helcl et al., 2018) or to a number of subword tokens in all sentences (e.g. Vaswani et al., 2018). The former definition correlates with measuring the number of training examples. However, the latter is optimized for training as it can better use the available GPU memory.

Another option is to measure the time passed to achieve a given result. Popel and Bojar (2018) use this reporting of wall-clock time and justify it by stating that time computed as steps per second, fluctuates at most by 2% during the training. Wall-clock time can be more informative than reporting a number of seen examples since the training step can contain a variable number of sentences or tokens. However, the training time is heavily influenced by the hardware and the other load on a given machine.[4]

Lastly, practitioners are primarily concerned with the error rate and the cost they need to pay to achieve that error level. It is referred to as a hardware cost (Shallue et al., 2018). This cost can be measured by multiplying the number of training steps by the average price per one step. It heavily depends on the respective hardware, but the number of training steps is hardware-agnostic and can be computed for any hardware given the average cost per step.

We run our experiments on various types of hardware. Our department's cluster contains three types of GPU cards: NVidia GeForce GTX 1080, the Titan version 1080Ti and Quadro P5000. We use the T2T framework, which uses batches of a varying number of sentences but approximately the same number of tokens. Thus reporting the training time is not comparable due to various machine setups and the number of examples or epochs is not exact due to the T2T batching behavior. We report the number of training steps, the actual updates of the NN.

[4]The NN research is usually carried out on clusters, where several people run various processes. Even when the GPU is allocated only for a given job, other shared resources, like CPU or network disks, can slow down the training process.

3.5.1 Convergence and Stopping Criterion

Training of NMT models is complicated by the fact that the learning curves, showing the performance of the model over the learning period on a fixed development set, usually never fully flatten or start overfitting on reasonably large datasets. Signs of overfitting[5] are noticeable in low-resource settings only, as discussed in Section 2.1.2. Especially with the recent models trained on a large parallel corpus, we can get some improvements, usually around tenths of a BLEU point, even after several weeks of training and the learning curve will not fully flatten out.

The common practice in machine learning is to use a stopping criterion. One option is to set the maximum number of training steps or epochs. The second option is to evaluate the model every X steps (or minutes) on the development set and stop the training whenever the last N updates do not improve the performance by at least some small delta.

The former approach relies on an intuition of how long approximately is enough to train the model. Usually, more complex models need more time to reach the maximal score, and an incorrectly set number of steps could stop the training prematurely. The latter approach is sensitive to the number of steps (and their duration) between individual evaluations: if the evaluations are too close to each other, the training can stop too early due to the training fluctuations, and when they are too far apart the training would not stop in a reasonable time on big datasets.

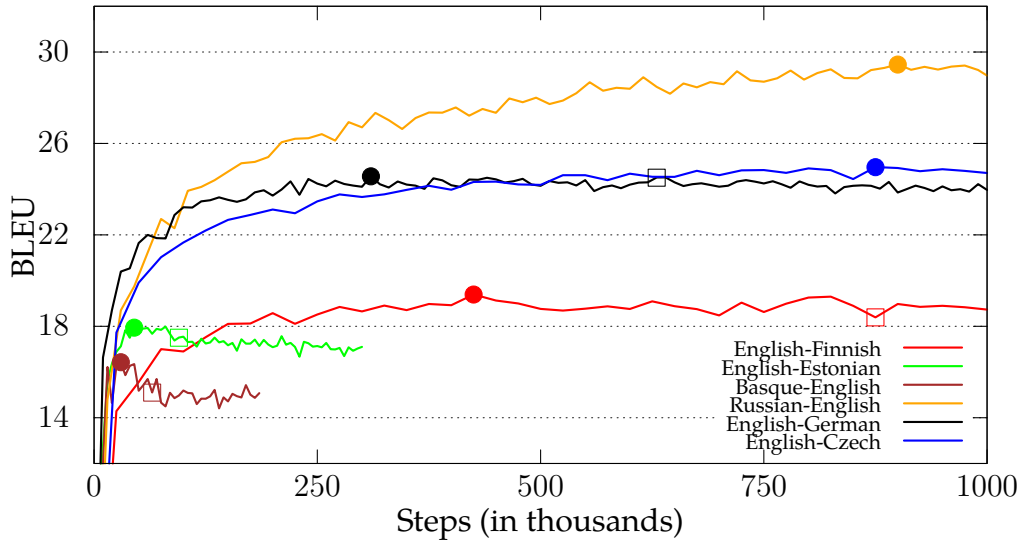
Many papers do not specify the stopping criteria or only mention an approximate time or the number of steps for how long the model was trained (Bahdanau et al., 2014; Vaswani et al., 2017). Presumably, the models are trained until no apparent improvement is visible on the development set. However, this stopping criterion is not perfect since the models could be stopped at various stages of training, and the comparison could be unfair.

In this thesis, we compare low-resource language pairs that converge within 50k steps and high-resource pairs that improve even after 1000k steps. Hence, to avoid premature stopping, we set a high upper level of maximal steps for each language pair. The low-resource languages, with less than 500k training examples, are trained for at most 200k steps. We also evaluate low-resource languages more often. The rest of the languages are trained at most for the 1000k steps.

However, this stopping criterion usually trains for a much longer time than necessary. Thus later in our work, we defined a more general convergence criterion. We stop the training earlier whenever there was no improvement bigger than 0.5% of maximal reached BLEU within the past 50% of evaluations. This criterion is comparable to stopping after X batch updates without any improvement, and it is less sensitive to the number of steps between evaluations as the low-resource languages are evaluated up to ten times more often. Importantly, we set mild conditions for stopping criteria on purpose to prevent the risk of premature judgments.

[5] By overfitting we mean the situation, when the performance on the training set is improving, but the score on the development set is worsening.

Figure 3.4: Examples of real learning curves. Full dots represent the best performance, squares represent our stopping criteria.



Our stopping criterion is especially useful for low and middle resource languages. Figure 3.4 presents with square a point where our training stops. Without any doubt, the model already passed its best performance, and thus the stopping is valid. We can notice that for high-resource languages, the stopping criterion does not trigger within 1000k steps.

With the stopping criteria in mind, once the training stops, we take the best performing model on the development set and report the number of training steps instead of the last step where the training stopped. Additionally, we regularly report full learning curves. Thus the readers can judge by themselves if they would expect any sudden change in the final performance.

4
Transfer Learning

Humans have the ability to utilize knowledge from previous experience when learning a new task. It helps us to learn new skills in a shorter time and with less effort. In fact, the more related a new task is, the faster we learn. In contrast, machine learning algorithms usually learn the task from random initialization on isolated data without any prior knowledge. Transfer learning attempts to change this approach by improving the performance on a new task with the usage of knowledge obtained for solving other tasks (Bahadori et al., 2014; Farajidavar et al., 2015; Moon and Carbonell, 2017).

Torrey and Shavlik (2010) describe three ways of how transfer learning can improve performance. Specifically:

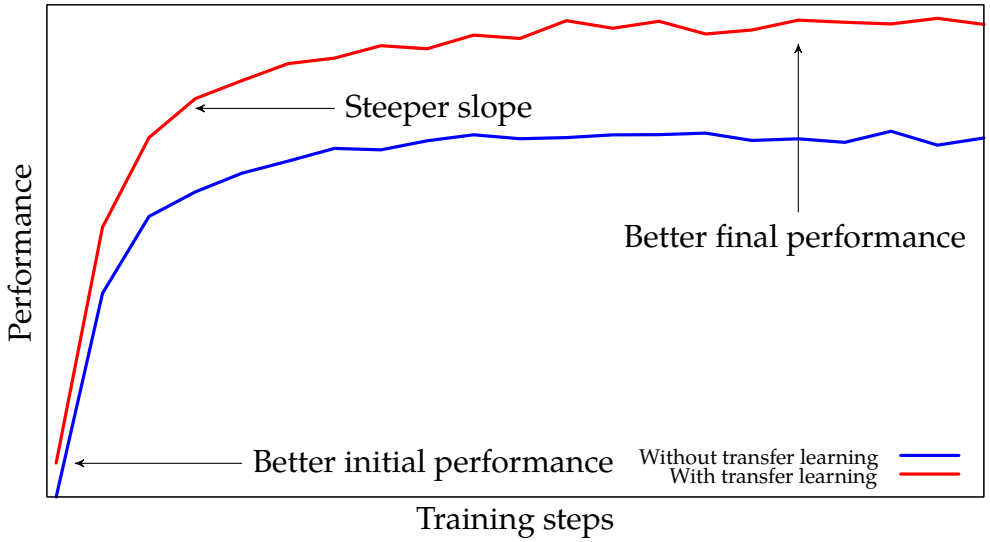
- improving the initial performance at the beginning of training compared to a randomly initialized model when the tasks are similar;
- shortening the time needed to reach the maximal performance;
- improving the final performance level compared to training the model without the transfer.

These three potential improvements are illustrated on learning curves in Figure 4.1.

The success of transfer learning is not always guaranteed. For example, when transferring from a weakly related task, it may hinder the final performance in the target task. A phenomenon known as the negative transfer has been well recognized by the transfer learning community (Pan and Yang, 2010; Wang et al., 2019). However, there is a lack of research on this phenomenon in the NMT field, which we investigate in Section 5.1.

In this thesis, we are highlighting various observations expressed in general way. However, we can only claim that these observations are general under the specific settings of the individual experiments. We are not claiming their decisive generality, because proving it across most language pairs, various experiment settings, and other conditions is out of the scope of this thesis.

Figure 4.1: Three impacts where transfer learning improves the training process. These are real learning curves for warm-start transfer learning on English→Estonian (see Section 4.7).



However, we are basing our observations on as broad a set of experiments as is reasonably possible. The list of all observations is at the end of the thesis in *List of Observations* on page 141.

Observation 1: *Observations in this thesis can be generalized only to the extent of our experiment settings.*

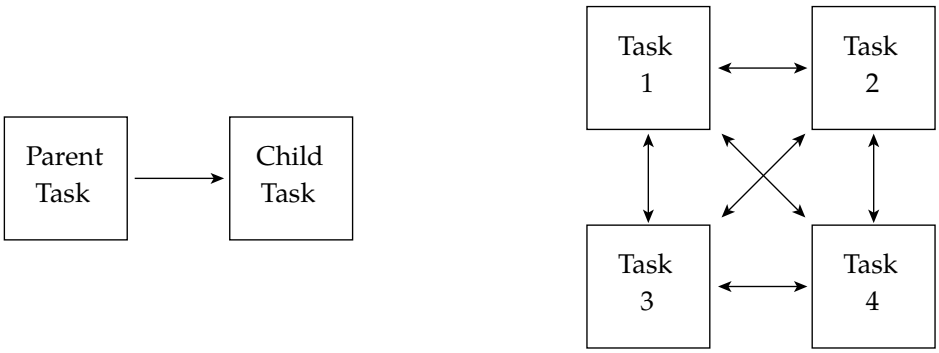
This chapter is organized as follows. First, we specify the terminology used across this thesis. In Section 4.2, we briefly describe the domain adaptation approach, which is a stepping stone for transfer learning. Then in Section 4.3 we dive into transfer learning, which is categorized into two groups. We investigate one of the groups in Section 4.4 and introduce two techniques discussed separately in Section 4.5 and 4.6. The second group is described in Section 4.7. We wrap up the chapter with a comparison of the proposed techniques in Section 4.8, followed by related work in Section 4.9 and the conclusion in Section 4.10.

4.1 Thesis Terminology

The main idea behind transfer learning is to pass on learned knowledge from one model to another. We denote the first model from which parameters are transferred as a *parent* and the designated model as a *child*.

In literature, we can often find a naming convention of *teacher* and *student*, however, it is more related to the knowledge distillation (Hinton et al., 2014), where the parent model (the teacher) is used to generate examples instead of directly sharing parameters. Another terminology is *source* and *target* tasks (Torrey and Shavlik, 2010), which is unsuitable for this thesis as we use these terms when referring to the source and target language of the language pair.

Figure 4.2: The information flows only in one direction from parent to the child task in transfer learning compared to the multi-task learning, where the information flows freely among all tasks improving them altogether.



Notably, in the NMT transfer learning it is customary to use the term "parent-child" (Zoph et al., 2016; Nguyen and Chiang, 2017), thus we are going to use it throughout this thesis.

Nonetheless, we use the naming convention of "parent-child" in this thesis for all scenarios, where there is a precedent model needed for training the second model regardless of the transferring technique. For example, during the backtranslation (see Section 5.6), the parent model generates training data for the child model, but no learned parameters are shared between models. Thus the child always specifies our designated task or the language pair for which we are trying to get the best performance. In contrast, we are not interested in the performance of the parent.

Both parent and child can use different language pair. For the parent model that translates from language XX to language YY (XX→YY), we recognize three scenarios:

- Shared-source language – a scenario where the source language is equal for parent and child. In other words, the child model translates XX→AA.
- Shared-target language – a scenario where the target language is equal, therefore the child model translates AA→YY.
- No-shared language – a scenario with no language shared, i.e. the child model translates AA→BB.

4.1.1 Multitask Learning

Multitask learning (Caruana, 1997) is closely related to transfer learning, where the goal is to solve several tasks simultaneously, as illustrated in Figure 4.2. Multitask learning differs from transfer learning as it needs to keep the performance of all tasks on a high level. In contrast, transfer learning can forget the knowledge of the parent task and only focuses on the performance of the child task.

It is possible to extend transfer learning to multiple parent tasks and therefore solve multi-task learning with techniques from transfer learning. However,

multitask learning is beyond the scope of this thesis, and we are not investigating it.

Transfer learning is a broad term used for various approaches not only in deep learning (Pan and Yang, 2010). In the domain of neural networks, and especially in the NLP, we can transfer knowledge either by transferring only a subset of trained parameters or the whole model.

When only the subset of model parameters are transferred, they are often the embeddings. We discussed transferring of pretrained embeddings in Section 3.1.3. Transferring all training parameter can be further split into transferring within the same task only by specializing the parent model or by transferring knowledge across different tasks. In MT, the former would correspond to the parent using the same language pair as the child, which is called *domain adaptation* described in Section 4.2. The latter case corresponds to different language pairs.

Despite transfer learning being such a broad term, it most often denotes transferring knowledge across different tasks. Thus we are going to use the term of transfer learning only in the scenario when we transfer all model's trainable parameters, and the parent and child use different language pair.

4.2 Domain Adaptation

Domain-specific MT systems are in high demand while general MT systems have limited applications. The general systems usually perform poorly, and thus it is important to develop MT for specific domains (Koehn and Knowles, 2017).

Domain adaptation is one of the critical issues in MT, where the goal is to specialize the model for a more specific domain. It is well known that an optimized model on a specific genre (news, speech, medical, literature, and other) obtains higher accuracy results than a generic system on the given domain (Gao and Zhang, 2002; Hildebrand et al., 2005). Specifically, when the training data have an unbiased distribution over the target domain, the final model will perform comparably on testing data as it performed during training on the development set. However, the performance will decrease if the training data comes from a different domain than the target domain (see Section 2.1.1). For example, when the training data are from news articles and the test domain is more specific as medical. Domain adaptation generally encompasses terminology, domain, and style adaptation.

We often have a large amount of out-of-domain parallel sentences. The challenge of training a domain-specific model is to improve the translation performance in the target domain, given only a small amount of additional in-domain data. This can be treated by *fine-tuning* (also called *continued training*) of the generic model with domain-specific data.

Domain adaptation has been successfully used in both statistical and neural MT (Gao and Zhang, 2002; Hildebrand et al., 2005; Luong and Manning, 2015; Pecina, 2017). In-domain gains have been shown with as few as dozens of in-domain training sentences (Miceli Barone et al., 2017).

In a typical NMT domain adaptation setup (Luong and Manning, 2015; Servan et al., 2016; Chu and Wang, 2018), we first train a parent model on a

resource-rich out-of-domain parallel corpus. After the model is trained on the general model, we exchange the training corpus for the in-domain one and follow by fine-tuning the parent model. We can view the domain adaptation as transfer learning from the out-of-domain parent model into the domain-specific child model.

Freitag and Al-Onaizan (2016) stated two problems of domain adaptation. First, it is prone to over-fitting due to the limited amount of in-domain data. Second, the process of fine-tuning the parent model on the in-domain data leads to deteriorating the performance on the general domain. Freitag and Al-Onaizan (2016) propose to tackle the problem by ensembling the parent model with the child model. They showed that the ensembled model improves on the in-domain testset and preserves its performance on the general domain. Chu et al. (2017) address these problems by mixed fine-tuning, where they combine the out-of-domain corpus with oversampled in-domain data before adapting the general model.

Most researchers investigating domain adaptation assume a scenario where the domain of the data is given, and the in-domain data exists. However, in real-life scenarios such as an online translation engine, the domain of the sentences is not given, and guessing the domains of the input sentences is crucial for proper translation. A method to address the lack of in-domain data can be tackled by classifying the domains of individual sentences in the training data followed by search and selection of training sentences close to the target domain (Farajian et al., 2017; Li et al., 2018).

This is only one of the approaches for domain adaptation, one which is closest to transfer learning. However, there are many more, especially in different tasks than MT (Bruzzone and Marconcini, 2009; Pan et al., 2010).

4.3 Transfer Learning

In the context of NMT, transfer learning as well as domain adaptation (see Section 4.2) refer to the situation with a mismatch between the train and test data distribution. In the case of domain adaptation, the parent and child operate on the same language pair but differ in the domain of data. In contrast, transfer learning uses a different parent language pair than the child model.

Since the domain adaptation works by fine-tuning the parent model trained on the same language pair, it does not require any modification in architecture nor the vocabulary and relies on the continued training process. In comparison, when we want to retrain the parent with a different language pair than the child, we need to tackle the problem with vocabulary mismatch between parent and child languages. There are two groups of approaches to solving the vocabulary problem based on their application either before training the parent model or after the training. We discuss these two categories in the next section.

4.3.1 Transfer Learning Categories

We distinguish two categories of transfer learning based on whether we have training data for the child language pair at the time of training the parent model. Neubig and Hu (2018) call the approaches warm-start and cold-start. In the

warm-start approach, we have the child training data available at the time of training the parent, and we can take steps to prepare the parent model for transfer learning. In contrast, the cold-start approaches use a general parent that has not been modified in advance in any way for the child language pair and apply different modifications before training the child model.

Intuitively, we expect the warm-start to perform better because it can better handle the child language pair to some extent. However, the additional training time of the child's specific parent model can be costly. For example, whenever researchers compare various language pairs or hyperparameter setting, they would need to train an individual parent for each such scenario and the time and hardware cost would considerably increase. Therefore, they can consider a scenario to train a strong general NMT system only once and use it repetitively in the cold-start transfer learning. Furthermore, there are situations where the ability to quickly learn the MT model given only a small amount of training data can be crucial. For example, when a crisis occurs in a region where people speak an under-resourced language, quick deployment of the MT system translating from or to that language can make a massive difference in the impact of the provided support (Lewis et al., 2011). In such a case, having a strong general NMT system available for a quick transfer learning of any low-resource language pair is a great advantage.

We use the same vocabulary whenever the parent and child have the same set of languages, i.e. disregarding the translation direction and model stage (parent or child). For example, we use the same vocabulary for English→Estonian model as well as Estonian→English although new vocabulary could be generated for each model separately.

Techniques and results in this chapter have been published mainly in two papers. The cold-start scenario is described in the paper Kocmi and Bojar (2019a), and warm-start scenario is investigated in the paper Kocmi and Bojar (2018). This chapter includes results as well as short textual parts from those two papers.

4.4 Cold-Start Transfer Learning

The main problem of transfer learning is the mismatch of parent and child vocabulary. The cold-start transfer learning tackles the problem after the training of the parent by modifying the parent model right before transferring parameters to the child model.

Whenever the parent model uses vocabulary with a high overlap with the child's vocabulary, we can ignore the differences and train the child with the parent's vocabulary. We call this approach "Direct Transfer", which we discuss in Section 4.5. The second option is to transform the parent vocabulary right before the child's training in various ways to accommodate the needs of the child language pair. Approaches from the second group are discussed in Section 4.6.

All cold-start approaches rely on the ability of neural networks to quickly adapt parent parameters to new conditions, i.e. segmenting words usually to more tokens than in parent model or remapping parent subwords embeddings to unrelated child subwords. We show that NMT can quickly adapt and obtain better performance on a given child language pair than by training from random initialization.

The experiments and short textual parts in this section are from our paper Kocmi and Bojar (2019a) and we use $T2T_8$.

4.4.1 Cold-Start Evaluation

The cold-start approach can be used with any parent model, even one trained by different researchers. In order to demonstrate that our cold-start methods are general, not restricted to our laboratory setting and that the results are easily replicable, we do not train the parent model by ourselves, but for all experiments regarding the cold-start transfer learning we reuse the winning model from the WMT 2018 Czech–English translation shared task trained by Popel (2018b). The model was trained with CzEng 1.7 and several other smaller corpora.

We decided to use this model for several reasons. It is trained to translate between English and Czech, a high-resource language pair where Czech is not similar to any of the languages we use.[1] It is trained using the state-of-the-art Transformer architecture as implemented in the popular Tensor2Tensor framework[2] (Vaswani et al., 2018).

We use two parent models: the English→Czech and Czech→English. Where the parent and child model always use English on the same side. For example, the English→Russian child has English→Czech as the parent.

To the best of our knowledge, we are the first to use transfer learning on a model trained by someone else, showing the proof-of-concept for recycling models in NMT, which reduces the overall training time of NMT. Note that reusing neural models is more common in other tasks (see Section 3.1).

4.5 Cold-Start Direct Transfer

Subword-based vocabulary can represent any text from any language by breaking the words down to characters or bytes.[3] We exploit this behavior that a parent vocabulary can encode any child's words and use the parent model as is. We call this approach the "Direct Transfer".

In the Direct Transfer approach, we ignore the specifics of the child vocabulary and train the child model using the same vocabulary as the parent model. We take an already trained parent model and use it to initialize parameters for a child language pair. In other words, we continue the training process of the parent model on the child trainset without any change to the vocabulary or hyper-parameters. This applies even to the training meta-parameters, such as the learning rate or moments.

[1] From our target language selection (see Section 2.3.2), the linguistically closest language is Russian, but we do not transliterate Cyrillic into Latin script. Thus the system cannot associate similar Russian and Czech words based on subword appearance.

[2] `https://github.com/tensorflow/tensor2tensor`

[3] The standard implementation of BPE segmentation by Sennrich et al. (2016b) cannot represent unknown characters by breaking them to bytes. However, the implementation could be extended to support encoding of bytes by escaping the byte representation in the same way as in the wordpieces.

Table 4.1: "Baseline" is a model trained from random initialization with own specific vocabulary. "Direct Transfer" is using the parent vocabulary. Models in the top part of the table use the English→Czech parent model, models in the bottom part use Czech→English. The scores and difference are in BLEU. The ‡ represents significantly better results based on significance test described in Section 2.4.3.

Translation direction	Baseline	Direct Transfer	Difference
EN→Odia	**3.54** ‡	0.04	-3.50
EN→Estonian	16.03	**20.75** ‡	4.72
EN→Finnish	14.42	**16.12** ‡	1.70
EN→German	36.72	**38.58** ‡	1.86
EN→Russian	**27.81** ‡	25.50	-2.31
EN→French	33.72	**34.41** ‡	0.69
French→Spanish	31.10	**31.55** ‡	0.45
Estonian→EN	21.07	**24.36** ‡	3.29
Russian→EN	**30.31** ‡	23.41	-6.90

This method of continued training on different data while preserving hyper-parameters is essentially a domain adaptation technique, where the domain is a different language pair.

The intuition behind the Direct Transfer is that NN is robust enough that the vocabulary mismatch can be disregarded altogether as long as there is some overlap between the child and parent vocabulary. This is mainly due to the usage of subword tokens, which segment any text into a sequence of allowed subwords (see Section 3.2). However, Direct Transfer suffers from a deficiency in the segmentation of child words, which can lead to splitting words to individual characters or even bytes that are hard for NMT to correctly translate.

Thanks to the simplicity of the Direct transfer, it can be easily applied to existing training procedures as it does not need any modification to the NN frameworks.

In the following sections, we discuss automatically assessed translation quality. We show the results of Direct Transfer in Section 4.5.1, followed by an analysis of the usage of parent vocabulary in Section 4.5.2 and introduction of a problem with vocabulary mismatch in Section 4.5.3. We conclude by the analysis of Direct Transfer drawbacks in Section 4.5.4.

4.5.1 Direct Transfer Results

We start with the results of Direct Transfer method, which uses parent vocabulary without any change. The results of our evaluation are tabulated in Table 4.1. In comparison to the baseline, the performance of Direct Transfer is significantly better in both translation directions in all cases except for Odia and Russian, which use a different writing script and we discuss it later in Section 4.5.4, where we show that it is primarily a problem of filtering long sentences.

Importantly, our baseline, trained only on child data, has an advantage over cold-start transfer learning as it uses child-specific vocabulary. Closer baseline to our transfer learning setup would be to use the parent vocabulary even for baseline, which would lead to an even larger difference in the performance. However, we decided to use the stronger baseline.

The Estonian–English pair confirms that sharing the target language improves performance as previously shown on similar approaches (Zoph et al., 2016; Nguyen and Chiang, 2017). Moreover, we show that the improvements are significant for the translation direction from English, an area of transfer learning neglected in previous studies.

The largest improvement of 4.72 BLEU is for the low-resource language pair English→Estonian. Furthermore, the improvements are statistically significant even for a high-resource language such as 0.69 BLEU increase for a high-resource English→French. To the best of our knowledge, we are the first to show that transfer learning in NLP can be beneficial also for high-resource languages.

Observation 2: *Direct Transfer can significantly improve the performance of the child model in both directions for both low-resource and high-resource language pairs.*

The basic intuition behind the improvements in translation direction into English is that the models reuse the English language model in the decoder and therefore the improvements are due to better ability to generate grammatically correct sentences without the context of the source language. Although better decoder's language model could be one of the reasons behind the improvements, it cannot be the only explanation since we see the improvements also for translation direction where English is the source side, and therefore the decoder has to learn a language model for the second language.

Furthermore, we get improvements even for child language pairs in the no-shared language scenario. In our study, we evaluated French→Spanish, which got a 0.45 BLEU improvement. Although, in this particular case, we need to take into account that it could be partly due to these languages being linguistically closer to the parent's source language English.

In Section 5.2, we discuss that the shared-target language is easier for NMT than the shared-source language. It is also the main reason why we compare more systems in the direction from English rather than to English.

The results of such performance boost are even more surprising when we take into account the fact that the model uses the parent vocabulary and thus splits words into considerably more subwords, which we carefully analyze in Section 4.5.3. It suggests that the Transformer architecture generalizes very well to short subwords and is robust enough to generate longer sentences.

In conclusion, the Direct Transfer learning improves the performance in all cases except English→Odia, English→Russian and Russian→English. In order to shed light on the failure of these languages, we need to analyze the parent vocabulary.

4.5.2 Parent Vocabulary Effect

The problem of OOV words is solved by using subwords segmentation at the cost of splitting less common words into separate subword units, characters, or even individual bytes, as discussed in Section 3.2. The segmentation applies

Figure 4.3: A toy example of using English wordpiece segmentation onto Czech sentence. For simplicity, we suppose the vocabularies contain all ASCII characters in addition to the tokens specifically mentioned.

Czech vocabulary:	{bude, doma_, end, me_, ví, vík}
English vocabulary:	{bud, dom, end, ho, me, week_, will}
Czech Sentence:	`O víkendu budeme doma.`
Segmented by Czech vocab.:	`O_ vík end u_ bude me_ doma_ ._`
Segmented by English vocab.:	`O_ v \ 2 3 7 ; k end u_ bud e me_ dom a_ ._`

deterministic rules on the training corpus to generate the subword segmentation that minimizes the number of splits for the observed word frequencies to fill up the vocabulary of a predefined size (see Section 3.2).

However, when using a subword segmentation created for a different language pair, the condition of the optimal number of splits is not guaranteed. Especially more linguistically distant languages that contain only a small number of common character n-grams need more splits per word.

The example in Figure 4.3 shows that using a vocabulary for a different language leads to segmenting words to substantially more tokens, especially in the case when the language contains characters not available in the vocabulary. This is most crucial as the unknown character is first transformed into byte representation ("\237;" in Figure 4.3) that is later handled as a standard text.

In Figure 4.3, English vocabulary doubles the number of tokens in the investigated sentence relative to the Czech vocabulary.

Direct Transfer approach uses the parent vocabulary, which can lead to segmenting the child training set into many individual tokens that could harm the MT performance. In order to study this effect, we examine the influence of using different vocabulary on the training dataset of the child.

We consider the parent Czech–English vocabulary (Popel, 2018b) used in our experiments and apply it for segmentation of language pairs and compare the average number of subwords per word. We examine the language pairs and their training sets that are used in experiments regarding Direct Transfer.

Table 4.2 documents the splitting effect of various vocabularies. When using the language-pair-specific vocabulary (column "Specific"), the average number of subword tokens per word (denoted "segmentation rate") is relatively constant for the English around 1.2 subwords per word as well as other languages except for the Odia language with 3.7 tokens per word, which we explain in Section 4.5.2. This regularity possibly emerges from the size of vocabulary and the number of words in both languages.

Observation 3: *Using a language-specific wordpiece vocabulary has a consistent segmentation rate around 1.2 subwords per word.*

If we use the Czech–English vocabulary on the child dataset (column "EN-CS"), there is an apparent increase in the average number of subword tokens per word. For example, German has twice as many tokens per word compared to the language-specific vocabulary that has 1.3 tokens per word on average.

Table 4.2: Average number of tokens per word (tokenized on whitespace) when applied to the training corpora. "Specific" represents the vocabulary created specifically for the examined language pair. "EN-CS" corresponds to the use of Czech–English vocabulary. The "First language" represents English, except of the last row, where it represents French. "Second language" represents the other language of a given language pair.

	First language		Second language	
Language pair	Specific	EN-CS	Specific	EN-CS
EN–Odia	1.2	1.4	3.7	19.1
EN–Estonian	1.2	1.2	1.1	2.3
EN–Finnish	1.2	1.2	1.1	2.6
EN–German	1.2	1.2	1.3	2.5
EN–Russian	1.3	1.4	1.3	5.3
EN–French	1.3	1.4	1.6	2.5
French–Spanish	1.3	2.1	1.2	2.1

Russian has four times more tokens per word due to Cyrillic, similarly for the Odia script.

Russian Cyrillic alphabet happens to be contained in the parent vocabulary together with 59 Cyrillic bigrams and 3 trigrams, which leads to 5.3 tokens per word. The Odia script is not contained in the Czech–English vocabulary at all, leading to the splitting of each word into individual bytes, which explains the 19.1 tokens per word (see Figure 4.3).

The first language is not affected by the parent vocabulary much (only slightly for the French-Spanish language pair) because English is shared between both the child and the parent vocabulary. The second language that differs between parent and child approximately doubles the number of splits when using parent vocabulary (see the difference between both columns "EN-CS").

Observation 4: *Wordpiece vocabulary roughly doubles the segmentation rate for different child languages that use the same script as parent languages.*

It is necessary to mention that the datasets have various domains and various sizes and therefore the average number of tokens could be different on various domains even for the same language pair. The size of the vocabulary[4] is also crucial as it defines the number of available subwords. Moreover, the length relation between the source and target sentences influence the final vocabulary.

The use of a different segmentation roughly doubles the number of tokens per sentence for languages using the same writing script. Therefore the NMT models that use Direct Transfer have to adapt to different sentence length in comparison to the parent. However, as we showed in Section 4.5.1, the Direct Transfer significantly improves the performance over the baseline showing the robustness of NMT.

[4]We use vocabulary with 32k subwords in all experiments.

Odia Subword Irregularity

We try to shed some light, why Odia has, on average, more tokens per word after subword segmentation even when using a language-specific vocabulary. The Odia script (also called Oriya script) has 52 characters, which lead to more character combinations than in English, which we believe is linked to a higher number of subwords per word.

To confirm our intuition, we investigate the Odia–English vocabulary. The average length of Odia tokens in the vocabulary is 4.2 characters compared to the 6.9 characters for English subwords in the same vocabulary. The average length of a non-segmented word in the Odia–English training set is 6.4 characters for Odia and 5.2 characters for English. With that in mind, Odia has on average longer words but uses shorter subwords than English, which leads to the substantially higher average number of tokens per word as reported in Table 4.2 in comparison to other languages.

This is mainly due to the size of vocabulary, which is not enough for the Odia–English language pair. Larger vocabulary would contain longer Odia subwords, thus would make the segmentation less frequent. This fact could be one of the reasons why the performance of Direct Transfer is worse than baseline as reported in Section 4.5.1. On the other hand, Odia is a low-resource language, and having large vocabulary would result in fewer examples per individual subwords in training data. We investigate the problem in Section 4.5.4.

4.5.3 Vocabulary Overlap

Direct Transfer uses parent vocabulary, and we showed how it increases the segmentation of the child's training corpus in Section 4.5.2. Now we examine what percentage of the parent vocabulary is used by the child language pair and investigate how large is the part of parent vocabulary that is left unused with the child language pair.

In order to find tokens from the parent vocabulary that are used by the child model, we segment the training corpus of the investigated languages with the Czech–English vocabulary and count how many unique tokens from the vocabulary appear in the segmented child's training corpus.

The percentage of used tokens are in Table 4.3. Before examining the results, we need to mention that the training sets are usually noisy, and some sentences from other languages can easily appear in them. For example, it is often a case that part or whole sentence is left untranslated in the parallel corpus. For the simplicity of our argument, we do not remove any foreign sentences nor ignore the least used tokens in any way. Therefore, the actual used part of parent vocabulary by a given language is smaller than presented in Table 4.3.

With that in mind, we can notice that English always uses more tokens from the vocabulary than the second language. This is caused by English being the shared language and already present in the vocabulary. Although the reverse is true for the original dataset of Czech–English where English occupies a smaller part of the vocabulary (71.1%) than Czech (98.0%). The reason why the total is not 100% but only 98.8% (see column "Both") is possibly due to a slightly different training set as the vocabulary was prepared by Popel (2018b).

Table 4.3: The percentage of the parent vocabulary tokens used in the child's trainset. The vocabulary is shared for both languages. The column "Both" represents the number of vocabulary tokens used by both languages.

Language pair	1st language	2nd language	Both
EN–Czech	71.1%	98.0%	98.8%
EN–Odia	23.3%	0.8%	23.9%
EN–Estonian	54.8%	39.7%	57.0%
EN–Finnish	59.2%	54.5%	60.6%
EN–German	58.4%	52.9%	59.9%
EN–Russian	68.6%	67.0%	71.4%
EN–French	64.6%	64.5%	65.5%
French–Spanish	55.1%	54.0%	57.1%

We can see that the Odia does not use many of available subwords, as concluded in the previous section. Interestingly and contrary to our previous findings, Russian utilizes a substantial part of the parent vocabulary. After a closer examination of the training corpus, we noticed that the Russian part contains many Czech and English sentences. When we counted only subword tokens that contain at least one Cyrillic character, the used part of vocabulary dropped to 0.3% for Russian confirming the previous findings with extremely high segmentation rate than other languages.

The most important result is that most language pairs use around 60% of parent vocabulary. This means that remaining tokens are left unused and only slow down the training and inference because the model has to calculate the softmax over the size of the vocabulary. Thus in Section 4.5.4, we propose a vocabulary transformation approach that overrides these unused tokens with child-specific ones.

Observation 5: *Child language pair uses roughly 60% of the available parent vocabulary tokens when sharing one language with the parent in Direct Transfer.*

4.5.4 Direct Transfer Drawbacks

In this section, we discuss the failure to improve the performance of child models of English→Odia, English→Russian and Russian→English translation pairs. Intuitively, Russian and Odia use a different writing script than English, and therefore the parent vocabulary segments them into many wordpieces as we discussed in Section 4.5.3, which possibly harms the NMT and fails to adapt to such long sentences fully. However, we need to take into account another possible explanation.

The framework T2T drops sentences from training corpus that are too long in order to allow bigger batch sizes, which lowers the training time. In our experiments, the threshold is set to 100 subwords, which is based on our previous findings (Kocmi et al., 2018b), where it was enough for most of the languages.

Table 4.4: The amount of training corpus removed by filtering long sentences with more than 100 subwords (lower is better). The column "Czech–English" shows results when parent segmentation is used. The "Child-specific" exploits vocabulary prepared for each language pair separately, the child-specific vocabulary has been used by the baseline systems.

Language pair	Czech–English (%)	Child-specific (%)
EN–Odia	98.6	12.0
EN–Estonian	8.7	0.0
EN–Finnish	12.6	0.2
EN–German	4.0	0.3
EN–Russian	58.1	1.7
EN–French	10.8	1.0
French–Spanish	13.9	0.1

NMT translates individual sentences with an average length of 20-30 words. Thus the average segmentation of 1.2 leads to an average length of a sentence under 50 tokens. However, the limit was set for a language-pair-specific segmentation, and thus, we investigate the influence of a higher segmentation rate.

We segmented the training data with both Czech–English and language-specific vocabulary and compared the percentage of filtered sentences by counting sentences with less than 100 tokens. The percentage of training corpus that has been filtered out is presented in Table 4.4.

As we have feared, the filtration has removed 98.6% of sentences from the Odia–English corpus and 58.1% from the Russian–English corpus. Additionally, other language pairs also have a notable drop in the total amount of training data. We note that this happens silently on the fly in the T2T framework.

Observation 6: *Corpus filtration based on the length of the sentences can drop a large part of training corpora due to the higher segmentation rate when using a parent vocabulary.*

On the defense of the limit, when using the child-specific vocabulary, the filtering does not remove many sentence pairs, and therefore, it is justifiable. However, the limit does not take into consideration the usage of a not-optimized vocabulary.

These findings mean that the Direct Transfer results have been negatively affected in comparison to the baseline, and therefore, the actual result could have been even better. Unfortunately, we had made this analysis after running all the experiments. Since the Direct Transfer had improved the results over the baseline for most cases, we decided to avoid rerunning all the experiments with a different limit. We re-run only the experiments with a higher threshold for filtering long sentences only for the three systems that scored worse than the baseline and used a different script than the parent.

We increased the threshold from 100 to 500 tokens per sentence. This filtering is already high enough to maintain most of the training corpus. Specifically, it

Table 4.5: Direct Transfer performance with increasing the filtering limit. The best performance (BLEU) is in bold. The ‡ represents significantly better results when compared to baseline.

Language pair	Baseline	Filter 100	Filter 500
EN→Odia	**3.54** ‡	0.04	0.26
EN→Russian	**27.81** ‡	25.50	27.19
Russian→EN	30.31	23.41	**30.49**

removes only 3.6% sentences from the Odia–English and 0.8% from Russian–English corpus. Increasing the number of tokens per sentence increases the nodes in the computation graph. In order to fit the same model on GPU, we run these experiments on our 16GB GPUs Quadro P5000. Results are in Table 4.5.

We see that with a higher filtering limit, the performance indeed improves for Russian–English, but not for English→Odia. However, the baseline is still outperforming the Direct Transfer in English→Odia and Russian→English. It is an indication that the model cannot adapt to sentences segmented to individual bytes. However, it can adapt to short n-grams of characters as in Russian.

Interestingly, Russian→English performs on par with the baseline, neither of the systems being significantly better than the other. This finding suggests that the model can adapt in the situation when the source language is segmented, but the target language is not. In this case, the target language is not affected by a high segmentation rate. Intuitively, it suggests that the shared-target scenario is easier for transfer learning. In other words, the neural network can utilize the language model of the parent and adapt it. We analyze it in more depth in Section 5.2.

Observation 7: *Direct Transfer does not improve performance when a child uses language with a different writing script.*

In conclusion, Direct Transfer improves the performance over the baseline as long as the segmentation due to vocabulary mismatch is reasonably low. An interesting question is if there is a threshold for segmentation ratio, which would predict if the model will improve over the baseline or if the Direct Transfer fails, or if the explanation is because of sentence length mismatch between source and target sentences. We leave these questions for future work.

In the following section, we discuss a vocabulary transformation technique, which avoids the segmentation problem and further improves the performance of cold-start transfer learning.

4.6 Cold-Start Vocabulary Transformation

Direct Transfer relies on the fact that we use subword units that can be used to encode any textual string. The pre-processing splits unseen words into several subwords, characters or possibly down to individual bytes. This feature ensures that the parent vocabulary can, in principle, serve for any child language pair,

Algorithm 1: Transforming parent vocabulary to contain child subwords and match positions for subwords common for both of language pairs.

```
Input: Parent vocabulary (an ordered list of parent subwords) and the
       training corpus for the child language pair.
Generate custom child vocabulary with the maximum number of sub-
 words equal to the parent vocabulary size;
for subword S in parent vocabulary do
    if S in child vocabulary then
        continue;
    else
        Replace position of S in the parent vocabulary with the first
         unused child subword not contained in the parent;
    end
end
Result: Transformed parent vocabulary
```

but it can be highly suboptimal, segmenting child words into too many subwords, where individual subword units do not contain relevant meaning. As expected, this is most noticeable for languages using different scripts (see the statistics in Table 4.2). Also, the child does not use the whole vocabulary leaving around 40% unused, which only slows down the training and inference process. In order to avoid both problems, the unused token positions could be reused to represent subwords more suitable for the need of the child language pair.

In this section, we propose a vocabulary transformation approach that changes unused subwords in parent's vocabulary with child-specific ones. We show that a straightforward replacement of subwords leads to significant improvements in performance.

NMT models associate each vocabulary item with its embedding. When transferring from the parent to the child, we can remap subwords and their assigned embedding as trained in the parent model without any modification to the architecture. The remapped subwords become associated with embeddings that initially behave as trained for the original subwords and the NMT has first to retrain them.

The algorithm for vocabulary transformation is explained in Algorithm 1.

4.6.1 Results with Transformed Vocabulary

Direct Transfer significantly outperforms the baseline, trained only on the child data, whenever the parent vocabulary does not segment the child training sentences into many tokens. In this section, we evaluate our Transformed Vocabulary approaches, which remaps unused parent subwords to useful child-specific ones.

First two columns of Table 4.6 are the same as in Table 4.1. We added a column with results of Transformed Vocabulary. We see "Transformed Vocabulary" delivering the best performance for all language pairs except English→Estonian,

Table 4.6: "Transformed Vocab" has the same setting as Direct Transfer but merges the parent and child vocabulary as described in Section 4.6. The structure is the same as in Table 4.1. The baseline uses child-specific vocabulary. The statistical significance ‡ is measured between Direct Transfer and Transformed Vocabulary.

Model	Baseline	Direct Transfer	Transformed Vocab
EN→Odia	3.54	0.04	**6.38** ‡
EN→Estonian	16.03	**20.75**	20.27
EN→Finnish	14.42	16.12	**16.73** ‡
EN→German	36.72	38.58	**39.28** ‡
EN→Russian	27.81	25.50	**28.65** ‡
EN→French	33.72	34.41	**34.46**
French→Spanish	31.10	31.55	**31.67**
Estonian→EN	21.07	24.36	**24.64**
Russian→EN	30.31	23.41	**31.38** ‡

significantly improving over "Direct Transfer" in most cases. The only exceptions are Estonian→English, English→French and French→Spanish, where neither of the systems is significantly better than the other, however, both of them are significantly better than the baseline.

Furthermore, it confirms that our Transformed Vocabulary successfully tackles the problem of Direct Transfer when the child language uses a different writing script such as English→Odia, English→Russian, and Russian→English.

Observation 8: *Transformed Vocabulary is not negatively affected by parent vocabulary segmentation.*

We see that cold-start transfer learning is not restricted to the low-resource scenario as it also improves high-resource language pairs: Finnish–English, German–English, Russian–English, and French–English.

Observation 9: *Our cold-start transfer learning improves the performance of both low-resource and high-resource language pairs.*

Interestingly, the cold-start transfer learning technique improves even the scenario with no-shared language, in this case French→Spanish.

Furthermore, the positive results imply that the neural network is robust enough to correct the randomly assigned child-specific embeddings and therefore reuse even more capacity of the parent model.

4.6.2 Various Vocabulary Transformations

Our Transformed Vocabulary technique assigns unmatched subwords mostly at random. However, there are many other variants. We propose several of them and evaluate them in this section.

We noticed that the vocabulary is structured in stages roughly based on the frequency of subwords in the corpora. This is due to the vocabulary creation

Table 4.7: Comparison of various approaches to replacing tokens in parent vocabulary.

Language pair	EN→Estonian	Estonian→EN
Frequency-based	20.27	23.32
Everything random	16.41	19.84
Unmatched random	**20.28**	22.45
Levenshtein distance	20.04	**23.66**

that adds less frequent subwords in stages until reaching the requested size of the vocabulary. Therefore, we call the technique from the previous section "Frequency-based".

In contrast to frequency, we can assign tokens at random. Either all of them or only unmatching tokens. We call the former approach "Everything random". It is when all subword tokens are first shuffled and then assigned at random. This approach does not match any tokens. Therefore the NMT needs to learn even tokens that have been used by the parent model. The latter approach is called "Unmatched random". It first assigns subwords that are in parent and child vocabulary. Then it assigns the remaining child tokens at random.

Last option is the assignment of subwords based on some distance. We select the Levenshtein distance, which measures the number of edits between two strings. The vocabulary created by this technique assigns subwords iteratively by increasing the allowed distance for assignment. In other words, it starts by assigning all matching subwords (distance 0), then subwords that have a distance of one edit, then two edits and so on.

The results of comparing various approaches to replacing tokens in parent vocabulary are presented in Table 4.7. All approaches reach comparable performance except the "Everything random" assignment. Thus we found no significant differences in the performance as long as subwords common to both the parent and child keep their embeddings, i.e. are mapped to the same index in the vocabulary. The subwords unique to the child vocabulary can be assigned randomly to the unused parent embeddings. A similar result was observed by Zoph et al. (2016). They show that the random assignment of words in their approach works as well as an assigning based on lexical similarity.

Observation 10: *It is necessary to preserve tokens shared between parent and child in the same place. The order of assignment for the remaining tokens does not play an important role.*

There is still plenty of space for experiments with more advanced techniques of vocabulary and embedding mapping, e.g. utilizing multilingual embeddings like Multivec (Bérard et al., 2016). We leave this for the future work in order to keep our approach as straightforward as possible and focus more on the analysis.

Table 4.8: The number of steps needed for a model to converge. We present the step where the model has the highest performance on the development step based on the stopping criterion described in Section 3.5.1.

Language pair	Baseline	Direct Transfer	Transformed vocab
EN→Odia	45k	47k	**38k**
EN→Estonian	95k	**75k**	**75k**
EN→Finnish	420k	**255k**	270k
EN→German	270k	190k	**110k**
EN→Russian	1090k	630k	**450k**
EN→French	820k	**660k**	720k
French→Spanish	390k	435k	**375k**
Estonian→EN	70k	**30k**	60k
Russian→EN	980k	**420k**	700k

4.6.3 Training Time

Lastly, we also evaluate the total training time needed for cold-start transferred models to reach the best performance. The times in Table 4.8 represent the number of steps needed to reach the best performing models from Table 4.6. The steps are comparable due to the equal batch size. However, due to the training fluctuations in performance, it is not possible to define the exact step when the model converged. Therefore, the results should be seen only as a rough estimate of the training time. We use the stopping criterion as defined in Section 3.5.1

Both transfer learning approaches converged in a comparable or lower number of steps than the baseline as we see in Table 4.8. The reduction in the number of steps is most visible in English→German and English→Russian, where we got to less than half of the total number of steps.

Observation 11: *Cold-start transfer learning converges faster than training from random initialization.*

As mentioned earlier, the results are only approximate. However, based on the broad range of our experiments where we compared nine language pairs using various scripts and various training corpora sizes we conclude that both approaches of transfer learning are faster than training the model from random initialization.

4.6.4 Conclusion on Cold-Start Transfer Learning

The common practice in NMT is to train models from random initialization, which makes NMT demanding in terms of training time and hardware resources. Especially in the deployment where multiple language pairs are used, or we do not have resources to train all language pairs.

We proposed and studied two cold-start methods of transfer learning, namely Direct Transfer and Vocabulary Transformation, which improve the child model language pair regardless of the original parent training languages.

We achieve better translation quality and shorter convergence times than when training from random initialization. We showed the results on several language pairs as well as both translation directions. Furthermore, we showed the improvements even for high-resource language pairs such as English→French. The improvements are significant even in the scenario with no-shared language, in our case, the French→Spanish pair.

Above all, we showed a proof-of-concept of reusing models trained by others, thus training the parent model is not necessary, and anyone can use a model trained by others for transfer learning. The usage of models trained by Popel (2018b) proves that we could not manipulate the parent model in any way for transfer learning, for example, by hyperparameter search or using modified parent training data.

We also showed the robustness of neural networks that works despite the adverse conditions of randomly assigned child-specific subwords to embeddings previously trained for parent language pair. NMT have the ability to quickly retrain pretrained embeddings and obtain even better performance compared to the Direct Transfer.

The transfer learning technique, as presented in this chapter, is not suitable for scenarios with various architectures like various sizes of matrices or different network layouts. Likewise, the size of the parent vocabulary restricts the size of a child's vocabulary. As we saw in the example with Odia–English, using a language-specific vocabulary of 32k subwords leads to a high segmentation rate of 3.7 (see Table 4.2), and therefore a vocabulary with more subwords would be necessary to lower the segmentation rate. Although in this case of low-resource language, it would create other problems as it is better to use smaller vocabulary when handling low-resource languages (Sennrich and Zhang, 2019).

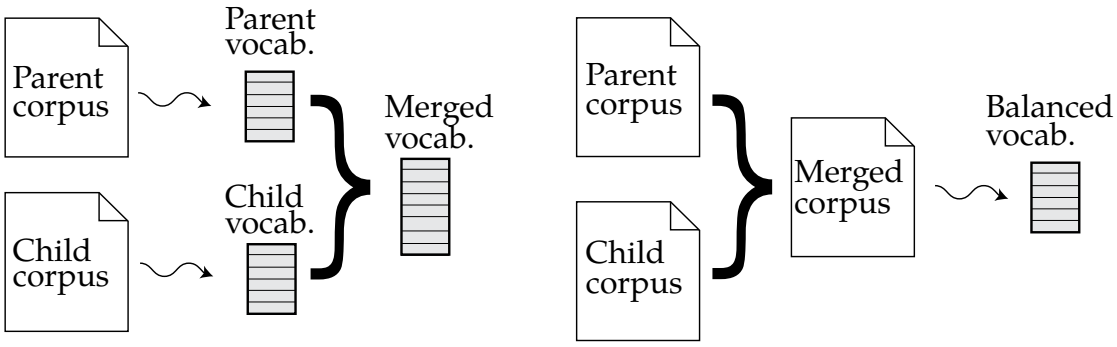
To conclude, the cold-start transfer learning does not need complicated modifications to the framework, only the ability to continue training. It improves performance and shortens the training time. Thus we suggest using transfer learning, especially the Transformed Vocabulary technique, as a new standard for model parameter initialization in scenarios where the architecture has not been changed, e.g. when not experimenting with various sizes of matrix dimensions.

In the following section, we investigate warm-start transfer learning, i.e. an approach that prepares the parent model with the knowledge of the child language pair in advance. In our case, we create a parent vocabulary already prepared for a specific child.

4.7 Warm-Start Transfer Learning

In the previous chapter, we discussed the cold-start scenario, where the parent model is trained in advance without prior knowledge about the child's language pair. The main disadvantage of that method is the need to reuse the parent vocabulary, which is associated with trained embeddings. The Direct Transfer, therefore, has a problem with a high segmentation rate and leaving roughly

Figure 4.4: A process of generation of shared vocabularies: Merged (left) and Balanced (right) Vocabulary.



40% of vocabulary unused. The Vocabulary transformation overcame these problems by randomly assigning child's subwords to unused embeddings, but this method restricted the maximal size of child vocabulary to be equal to the parent's vocabulary. Furthermore, the randomly assigned child subwords embeddings first had to be retrained. The warm-start scenario allows us to overcome such problems by preparing the parent model in advance for the upcoming transfer learning to the child's language pair.

The basic idea of how to avoid problems with randomly assigned embeddings is to directly use child-specific vocabulary at the time of parent model training. The use of a child's vocabulary in the parent model would inevitably undermine the performance of the parent model because we restrict the parent vocabulary. We studied this effect on child models in Section 4.5.2, where we showed that inappropriate vocabulary segments text to an increased number of tokens. However, in the transfer learning task, we do not pay attention to the final performance of the parent language pair. Thus we can ignore this performance drop.

The experiments in this section are mostly from Kocmi and Bojar (2018) and use $T2T_4$ if not specified otherwise.

4.7.1 Warm-Start Methods

In addition to using the child-specific vocabulary, we propose other variants of shared vocabulary in order to find the best approach for transfer learning. So far, we described two options: either to use parent-specific vocabulary (Direct Transfer) or child-specific vocabulary. Each of them has similar problems, but either for the parent or the child model. A solution is to create a vocabulary that is shared between both parent and child language pairs.

We propose two approaches of creating a shared vocabulary. One option of a shared vocabulary containing both the parent as well as the child subwords can be created by merging their specific vocabularies. We call this approach Merged Vocabulary. The second option is to utilize the algorithm for subword generation (wordpiece or BPE) by applying it to the concatenated corpus of both parent's as well as the child's vocabulary. This way, it prepares a balanced

vocabulary based on the subword frequency in both corpora. We call the second approach to Balanced Vocabulary. Both approaches are illustrated in Figure 4.4.

Merged Vocabulary is related to the cold-start Transformed Vocabulary as it tries to combine two separate vocabularies into one. The process of vocabulary generation in Merged Vocabulary is as follows: first, we generate vocabulary for both parent and the child language pair separately, and then we merge both vocabularies and remove duplicated subwords. Because the vocabularies are text files where each subword is on an individual line, the merge operation is a simple concatenation of both vocabularies together. Therefore this approach applies only to wordpiece-type vocabulary, which is not dependent on the ordering of the vocabulary such as the BPE algorithm.

The second approach, Balanced Vocabulary, is obtained by concatenating both parent's and child's training corpora into one corpus, which is then used to generate the vocabulary by the wordpiece (possibly also BPE) algorithm. Note that this corpus is used only for the vocabulary generation purposes and we are not using it for training NMT models.

The Balanced Vocabulary method is sensitive to the total size of the corpus. Whenever one of the corpora is bigger, more subwords of that language pair get into the final vocabulary. Mainly due to our focus on low-resource language pairs, we try to balance the amount of language-specific subwords in the vocabulary by randomly selecting an equal number of sentences from both corpora instead of using all sentences. Thus the mixed corpus contains 25% sentences from each of the four languages. We need to note that we do not take into account the number of words per sentence as it differs across corpora and also languages. Thus we suppose that corpora are usually already segmented on a sentence level and we disregard the average length of the sentences. We leave for future work an investigation of different ratios between languages and balancing the number of words instead of sentences across corpora.

Whenever we obtain the vocabulary by either of the methods, we follow the transfer learning pipeline that differs from cold-start by the fact that we need to train the parent model for each child. The training is the following: we train the parent model with a given vocabulary until convergence, followed by continued training on the child dataset. We remind the reader that in the case of warm-start transfer learning, we do not change the vocabulary nor any other (hyper)parameters when we exchange the training corpora. Therefore the NN is not forced to completely retrain embeddings when we change the training corpora from parent to child. It avoids the problem with the cold-start Transformed Vocabulary approach. Although some subwords have a different meaning in different languages, hence they need to be retrained even in the warm-start scenario.

4.7.2 Comparison of Warm-Start Techniques

In this section, we compare the two warm-start techniques proposed earlier with parent-specific and child-specific vocabularies to figure out which one leads to the best performance.

We compare the proposed methods using both pairs with both directions (i.e. two low-resource language pairs, namely Estonian–English and Basque–

Table 4.9: Results of various warm-start transfer learning approaches. The first two columns show the performance of the parent model, the third and forth column is the child model based on the corresponding parent in the same row. The baseline does not use transfer learning and uses language-pair-specific vocabulary. Scores are in BLEU and are comparable only within columns.

	Parent score		Child score	
Method	EN→CS	CS→EN	EN→Estonian	Basque→EN
Direct Transfer	22.78	27.81	15.55	23.29
Child-specific voc.	21.05	24.93	15.73	22.92
Balanced Vocabulary	22.58	**27.93**	**16.41**	**23.63**
Merged Vocabulary	**22.68**	✖	16.05	✖
Baseline	–	–	8.70	19.09

English). In this comparison, we use the parent model that has English on the same side. Thus English→Czech parent is used for English→Estonian and Czech→English parent is used for Basque→English.

We compare four different approaches: parent-specific (Direct Transfer), child-specific, Merged Vocabulary, and Balanced Vocabulary. All setups use the same layout of transfer learning and training conditions. We trained four parent models for English→Czech and four parent models for Czech→English, each with a different vocabulary. The parent training took one million steps before the transfer learning of the child.

The setups for Basque→English are from Kocmi et al. (2018c). We extended the evaluation with English→Estonian translation direction, for which we downsampled the original Estonian–English (see Section 2.3.2) corpus to only 100k sentence pairs. Hence, we artificially created a less resourceful language pair. The English→Estonian is based on different version of T2T framework than the rest of the section, namely $T2T_{11}$.

Merged Vocabulary approach generates a larger vocabulary because merging two vocabularies of equal size leads to a bigger size of the merged vocabulary. Moreover, larger vocabulary leads to less comparable results as improvements could be attributed to different vocabulary size. We want to evaluate all methods in the closest setting as possible. Thus for Merged Vocabulary, we generate smaller parent and child-specific vocabularies in the first place in a way that the final size after merging and removing duplicates is approximately the same as vocabulary size of other methods, in our experiments 32k subword vocabulary. Consequently, we define the size of initial two vocabularies experimentally by iteratively decreasing their size and measuring the size of merged vocabulary until we obtained the size of merged vocabulary within a 1% tolerance of the vocabulary size (the 32k subwords). The tolerance is in accordance with the T2T specific wordpiece implementation, see Section 3.2.2.

We start by investigation of the child model performance (column Child score) in Table 4.9. All four transfer learning techniques outperform the baseline trained only on the child language pair, the gains of nearly 8 BLEU points for

English→Estonian and 4 BLEU points improvement for Basque→English are significant.

Observation 12: *Both Balanced and Merged Vocabulary warm-start techniques significantly outperform the baseline.*

The missing result of Merged Vocabulary in Basque→English is because we have not conducted a comparison of this technique in Kocmi et al. (2018c).

Both Direct Transfer, or child-specific vocabulary setups, performed worse than the Merged and Balanced Vocabulary. We assumed that the parent-specific vocabulary would perform the worst since it introduces segmentation problems into the child language pair, as discussed in Section 4.5.2. Interestingly, the child-specific vocabulary is not the best approach even though the vocabulary is specifically tailored to the child model. Furthermore, the parent score is significantly lower (1.53–1.73 BLEU) for the child-specific vocabulary when compared to other variants, which is due to a sub-optimal vocabulary. This suggests that the translation quality of the parent model plays an essential role in the transfer model. This is an exciting result, as we usually disregard the performance of the parent in the transfer learning setup. We study the parent performance and behavior in Section 5.2.3.

Observation 13: *Child-specific transfer learning performs worse than Balanced or Merged Vocabulary even though its vocabulary contains more child-specific subwords.*

Lastly, we want to briefly discuss the ratio of parent vs. child subwords in the variants of the vocabulary. The parent-specific vocabulary (Direct Transfer) in general contains no child language pair subwords. On the other hand, child-specific vocabulary contains only child-specific subwords. In between are both Merged and Balanced Vocabulary that contain roughly half of the parent's subwords and half of the child's subwords. However, it could be the case that the best approach would be different, for example, including only 30% of parent vocabulary and 70% of child vocabulary. If that would be a case, it is most likely going to be specific for the parent and child language pairs and not being general for transfer learning. We leave this as an open question for future work.

In conclusion, both Merged and Balanced Vocabulary are better than baseline and the cold-start Direct Transfer. Furthermore, Balanced Vocabulary leads to better quality than Merged Vocabulary. However, we are not claiming that Balanced Vocabulary is strictly better than Merged Vocabulary. Lastly, the Merged Vocabulary technique is less general as it works only for the wordpiece segmentation (see Section 4.7.1). Therefore, in the following experiments and analysis regarding the warm-start transfer learning, we are going to use only the Balanced Vocabulary technique as the standard technique. We leave the Merged Vocabulary as an alternative for other work.

4.7.3 Broader Evaluation

In order to prove the generality of the warm-start method, we evaluate it across various parents and children, comparing linguistically related and unrelated languages as well as having shared English on the source or target side. We use language pairs described in Section 2.3.2.

Table 4.10: Transfer learning with English reused either in source (encoder) or target (decoder). The baselines correspond to training on one corpus only. BLEU scores are always reported for the child language pair. The scores are comparable within lines or whenever the child language pair is the same. The ‡ represents significantly better results.

		Balanced	Baselines: Only	
Parent	Child	Vocabulary	Child	Parent
EN→Finnish	EN→Estonian	**19.74** ‡	17.03	2.32
EN→Russian	EN→Estonian	**20.09** ‡	17.03	0.57
EN→Czech	EN→Estonian	**20.41** ‡	17.03	1.42
Finnish→EN	Estonian→EN	**24.18** ‡	21.74	2.44
Russian→EN	Estonian→EN	**23.54** ‡	21.74	0.80
EN→Czech	EN→Slovak	**17.75** ‡	16.13	6.51
Czech→EN	Slovak→EN	**22.42** ‡	19.19	11.62

As mentioned earlier, we are going to discuss only the Balanced Vocabulary approach ignoring the Merged Vocabulary and the child-specific vocabulary. Thus, whenever we talk about warm-start technique, we consequently mean the Balanced Vocabulary technique.

Table 4.10 summarizes our results for various combinations of a high-resource parent and a low-resource child language pairs. All comparisons are with the English on the same translation side for both parent and child. The baselines models are trained exclusively on the child or parent parallel corpus. We do not report parent score on parent testset.

The column with the child baseline is essential as it shows the impact of transfer learning. We see that for all language pairs, the transfer learning significantly outperform the baseline. However, as we evaluate some linguistically related languages, for example, Czech and Slovak, we also evaluate the performance of parent model only on the child's testset to show that without transfer learning the performance is strictly worse. For the Czech and Slovak, the parent alone can roughly translate given sentences and obtain 6.51 BLEU score for direction into Slovak and 11.62 BLEU for Slovak→English. In contrast, Estonian and Finnish are not as related as Czech and Slovak. Thus the parent model does not perform well on the child testset, obtaining only 2.44 BLEU for Estonian→English translation. This highlights that the improvement brought by our method cannot be solely attributed to the relatedness of languages.

Earlier works on NMT transfer learning (Dabre et al., 2017; Nguyen and Chiang, 2017) supposed linguistically related languages. We confirm their results also with our warm-start transfer learning on linguistically similar Finnish/Estonian and Czech/Slovak languages. Furthermore, the improvements are not limited only to related languages as Estonian and Finnish. Unrelated language pairs like Czech/Estonian or Russian/Estonian work comparably well.

Table 4.11: Comparison of various approaches for incorporating the child data into the parent trainset. All scores are in BLEU, and neither model is significantly better than any other.

Child model	No-Mixing	Mix with tag	Mix without tag
English→Estonian	**20.1**	**20.1**	19.9
Estonian→English	23.4	**23.7**	23.6

Observation 14: *Warm-start transfer learning improves performance even for unrelated languages.*

The most surprising is the comparison of English→Estonian performance across various parents. We see that Finnish, the linguistically related language, improves the performance the least compared to other parents. We reach an improvement of 3.38 BLEU for English→Estonian when the parent model was English→Czech, compared to an improvement of 2.71 from English→Finnish parent. This two improvements are statistically significant and differ from the conclusion of Zoph et al. (2016), who concluded that the more related the languages are, the better transfer learning works. We see it as an indication that the size of the parent training set is more important than relatedness of the languages as the Czech has 40.1M sentences and the Finnish only 2.8M sentences.

The results with Russian parent for Estonian child (both directions) show that transliteration is not necessary for our approach as used in previous works (Nguyen and Chiang, 2017). Therefore there is no vocabulary sharing between Russian Cyrillic and Estonian Latin (except numbers and punctuation, see Section 4.7.5 for further details), the improvement could be attributed to better coverage of English; an effect similar to domain adaptation.

We show that our method also works with the shared English on the source side. Although it is an intuitive result, we have to point out that earlier works focused only on the scenario with shared-target language (Zoph et al., 2016; Nguyen and Chiang, 2017). Similarly to cold-start, it supports the idea that improvements are not due to the better decoder's language model.

4.7.4 Combining Parent and Child Trainset

In the warm-start transfer learning scenario, we could train the parent model on a mixture of parent and child training data and then fine-tuning solely on child training data. We experiment with this setting to find out if the child performance is influenced by it.

In multilingual learning, Johnson et al. (2017) proposed to use a unique tag (separate word) "<2lang>" to identify desired target language. By adding this tag to each source sentence, they could train NMT model on a mix of training corpora. Another option is to mix the training corpora without any unique tag.

In the following experiment, we compare Balanced Vocabulary approach with three parent models trained on various trainsets: a standard no-mixing

Table 4.12: Performance in BLEU of a parent model evaluated on the child Estonian–English testset. In brackets are results evaluated on individual child models from Table 4.11.

Parent model	Mix with tag	Mix without tag
English→(Czech+Estonian)	**17.7 (20.1)**	2.4 (19.9)
(Czech+Estonian)→English	**22.0 (23.7)**	21.8 (23.6)

parent-only, a mix of parent and child with added tag, a mix of parent and child *without* a tag.

We use the Czech–English language pair as a parent and the Estonian–English as a child. We use $T2T_{11}$ setting.

In Table 4.11, we see that mixing corpora is slightly better for shared-target scenario. However, neither of approaches is significantly better than the others. Hence we see no difference between approaches. Although, this experiment could be influenced by the size of training corpora of the parent and the child.

The previous experiment showed that the performance is not largely influenced whether we mix the child data into the parent training data or not. However, in that setting, the parent can perform translation of the child language pair, because it was trained together with the child trainset. Thus we evaluate four parent models, trained on the mix of corpora from the previous experiment, on the child testset.

The performance of individual parent models is reported in Table 4.12. All of them perform significantly worse than after transfer learning of the child model. However, for the scenario, where the target language is shared between Czech and Estonian pair, or when the tag marks the target desired language, it is interesting that the models' performances are only slightly lower than for transfer learning than we would expect.

4.7.5 Balanced Vocabulary Analysis

Our Balance Vocabulary method relies on the vocabulary estimated jointly from the child and parent model. In the Transformer, the vocabulary is typically shared by both the encoder and the decoder. We analyzed the vocabulary in our cold-start scenario in Section 4.5.3, where we found out that around 40% of parent vocabulary is unused after the transfer.

Balanced Vocabulary is prepared in the following way. We take an equal part of parent and child corpus and generate a wordpiece vocabulary that is used both by the parent as well as the child model. With a large overlap, we could expect a lot of "information reuse" between the parent and the child.

We take the vocabulary of subword units as created for Russian–English parent and Estonian–English child experiments. This vocabulary contains 28.2k subwords in total. We then process the training corpus for each of the languages with this shared vocabulary, ignore all subwords that appear less than 10 times in each of the languages (these subwords will have little to no impact on the result of training) and break down the total of 28.2k subwords into overlapping

Table 4.13: Breakdown of subword vocabulary of experiments involving Russian–English parent and Estonian–English child.

Estonian	English	Russian	% Subwords
✓	-	-	29.93
-	✓	-	20.69
-	-	✓	29.03
✓	✓	-	10.06
-	✓	✓	1.39
✓	-	✓	0.00
✓	✓	✓	8.89
Reused parent			41.03

classes based on the languages where the particular subword was observed, see Table 4.13.

We see that the vocabulary is reasonably balanced, with each language having 20–30% of subwords unique (see the first three rows of Table 4.13). English and Estonian share 10% subwords not seen in Russian while Russian shares only 1.39% and 0% of subwords with each of the other languages, possibly due to the Cyrillic script. Overall, 8.89% of subwords are used in all three languages.

A particularly interesting subset is the one where parent subwords are used by the child model. In other words, subwords appearing anywhere in English and also tokens common to Estonian and Russian. For this set of languages, this amounts to 20.69+10.06+1.39+0.0+8.89 = 41.03%. We list this number on a separate row in Table 4.13, as "Reused parent". These subwords get their embeddings trained better thanks to the parent model. However, the vocabulary also contains 29.04% subwords used only by the parent and unused by the child.

Table 4.14 summarizes this analysis for several language sets used in the warm-start experiments, listing what portion is shared by all the languages (column "In All"), what portion of subwords benefits from the parent training (column "Reused from Parent") and what portion of vocabulary is unused by the child (column "Unused by Child).

We see a similar picture across the board; roughly 30% of subwords are unused by the child model. And roughly 50% of subwords are unused whenever both parent languages are distinct from the child. We remind that in the Direct Transfer the number was around 40%. We already discussed this problem as these subwords only slow down the training and inference as they are useless to the child.

Observation 15: *Balanced Vocabulary has 30% of tokens unused by the child in the vocabulary whenever one language is shared between parent and child.*

We list vocabularies used in our main results in Section 4.7.3 as well as language pairs not containing any shared language between parent and child (English in our case) with the child as we report in Section 5.1.4.

Table 4.14: Summary of vocabulary overlaps for the various language sets. The first column specifies what is the parent language pair. The child language pair is Estonian–English for all rows. All figures represent percentage of the vocabulary.

Languages	In All	Reused Parent	Unused by Child
Finnish–EN	19.5 %	49.4 %	26.2 %
Russian–EN	8.9 %	41.0 %	29.0 %
Czech–EN	20.3 %	49.2 %	21.2 %
Arabic–Russian	4.6 %	6.2 %	56.3 %
Spanish–French	18.4 %	34.1 %	28.7 %
Spanish–Russian	6.0 %	21.4 %	45.8 %
French–Russian	6.3 %	23.1 %	44.3 %

The Arabic-Russian-Estonian-English stands out with the very low number of subwords (6.2%) available already in the parent, mainly due to the scripts of parent language not using Latin. The parent Arabic-Russian thus offered very little word knowledge to the child, and yet it leads to a performance gain (21.74 vs. 22.23 BLEU, see Section 5.1.4).

Our observations indicate that the key factor is the size of the parent corpus rather than vocabulary overlaps. However, the reasons for the gains are yet to be explained in detail.

4.8 Warm-Start and Cold-Start Comparison

We study two approaches for transfer learning that differ in how the parent model is treated, specifically by allowing modification to the parent vocabulary before or after the training. In this section, we compare both cold-start approaches: the cold-start Direct Transform and Transformed Vocabulary with the warm-start approach of Balanced Vocabulary.

We train four parent models for warm-start approach differing in the vocabulary: two English→Czech and two Czech→English on the same parent training data. For the cold-start approaches, we used the models trained by Popel (2018b) similarly as in cold-start experiments (see Section 4.4).

Results are from Kocmi and Bojar (2019a) and we use $T2T_8$.

The results are tabulated in Table 4.15. We see that the warm-start method reaches in most cases significantly better performance in both directions. The only exception is the Russian→English and English→Estonian, where neither warm-start Balanced Vocabulary nor the Transformed Vocabulary is significantly better than the other.

The cold-start transfer learning improves the performance of high-resource language pairs (see Observation 9), and the warm-start improve the performance over the cold-start transfer learning. Hence warm-start should work for high-resource language pairs. This is confirmed by the Russian–English language

Table 4.15: The scores (BLEU) for cold-start methods (Direct Transfer and Transformed Vocabulary) with the warm-start method of Balanced Vocabulary.

Child language pair	Direct Transfer	Transformed Voc.	Balanced Voc.
EN→Estonian	**20.75**	20.27	20.62
EN→Russian	25.50	28.65	**29.03** ‡
Estonian→EN	24.36	24.64	**26.00** ‡
Russian→EN	23.41	**31.38**	31.15

pair in Table 4.15 and Slovak–English in Table 4.10. We study it further in the next Chapter 5.

Observation 16: *Both cold-start and warm-start transfer learning improve performance of low-resource and high-resource language pairs.*

The better performance of the warm-start approach is understandable since the parent model is already trained with vocabulary prepared to accommodate a given child language pair. It does not have the high segmentation problem as the Direct Transfer has, and it does not have to retrain randomly assigned embeddings as in Transformed Vocabulary.

However, the warm-start approach has one disadvantage over the cold-start: we cannot reuse *any* parent model, and we have to train the parent model for each child language pair separately. With that in mind, we investigate the number of steps needed to reach the performance as presented in Table 4.15.

Table 4.16 shows the total number of training steps. The cold-start scenarios show only steps needed for the child model convergence since we did not train the parent model by ourselves as we are using the model by Popel (2018b). In contrast, the steps for Balanced Vocabulary show the total number of steps the parent and the child were training altogether.

By observing the results, we see that due to the parent training, the warm-start scenario takes more steps to train. However, the total training time of the parent can vary broadly. Furthermore, in Section 5.4.6, we show that even a parent model that did not fully converge is a good parent for transfer learning.

In conclusion, the cold-start method has the advantage of not requiring to train the parent model, it does not need any modification to the training workflow (in Direct Transfer) or only a trivial modification of vocabulary (in Transformed Vocabulary) and reaches a better performance than training a baseline model. This makes it the candidate to most of the current training workflows, where researches could start using any model as a parameter initialization instead of random initialization as it is the current standard.

On the other hand, whenever the time is not a relevant criterion, but we are focused on the performance of the model, the warm-start scenario would be the recommended approach.

Table 4.16: Comparison of the number of steps needed for cold-start and warm-start methods to converge.

Child language pair	Direct Transfer	Transformed Voc.	Balanced Voc.
EN→Estonian	75k	75k	735k
EN→Russian	630k	450k	1510k
Estonian→EN	30k	60k	700k
Russian→EN	420k	700k	1465k

4.9 Related Work

The idea of transferring knowledge between different algorithms is as old as machine learning itself, and researchers were trying to reuse previously trained features on different tasks. Pratt et al. (1991) published a paper properly formulating the transfer learning problem by pretraining a neural network on a different task.

Firat et al. (2016) studied multitask learning, a scenario closely related to transfer learning. They propose multi-way multilingual systems, a generalization of work by Dong et al. (2015), with the primary goal of reducing the total number of parameters needed to cater multiple source and target languages. The system uses individual encoders and decoders for each language, with only the attention mechanism shared. The model is trained on one language pair at a time, thus to keep all the language pairs active in the model, a special training schedule is needed. Otherwise, "catastrophic forgetting" would hinder the ability to translate among the languages trained earlier. Firat et al. (2016) experimented with six languages, where the training data were between English and one of the remaining five languages. They showed that the multilingual set-up with shared attention outperforms the single-pair baseline on low-resource languages in all translation directions.

Nevertheless, when using high-resource languages with more than two million training sentences, experiments yield counter-intuitive results. The multi-way system outperforms the single-pair baseline whenever English is on the target side. However, whenever English is on the source side, the baseline performs on-par or better than the multi-way. The authors did not provide any explanation for this phenomenon. We discuss it in more depth when evaluating our results in Section 5.3. We believe that the improvements only for direction into the English are mainly due to the higher amount of sentences the English decoder had access to during the training in comparison to single-language-pair baseline. Therefore, the model can generate better English sentences. Firat et al. (2016) suggest that transferring knowledge from the language shared between parent and child is a harder scenario than transferring with the shared language. We analyze it in Section 5.2.

Johnson et al. (2017) introduce another multilingual approach. They add a special word (or token) to each sentence, which indicates the desired target language. Then they mixed together all parallel sentences from all language pairs together. Thus the NMT could recognize a target language by the token

"<2lang>" at the beginning of each sentence. The architecture then shares all parameters between all languages, not only the attention as in Firat et al. (2016). The model implicitly learns translation between all languages. However, it must be noted that the performance of their system is worse than the single-pair baseline. This is due to the same amount of parameters as in the baseline that is used between several language pairs, thus effectively having less amount of parameters for each used language. Johnson et al. (2017) support the hypothesis by increasing the size of the multilingual model, which reduces the performance gap to the baseline. The most interesting result is that the model can perform zero-shot translation, i.e. translating between languages never seen in the training set in a language pair.

There are relatively few studies investigating transfer learning in the area of the NMT. The reasons could be due to the brief history of NMT, which beginning is dated to the work of Sutskever et al. (2014), who presented the first end-to-end NMT model and it took until 2016 it became a major paradigm in MT (Bojar et al., 2016).

Zoph et al. (2016) claims to be the first to apply transfer learning to the low-resource NMT successfully. They used the word-level RNN translation model (Sutskever et al., 2014) with separate vocabularies for the source and target language. Their use shared English only in the scenario of shared-target. They exploit the cold-start scenario, where in contrast to our work, their approach required freezing the English embeddings from updating during the child's training. As for the second language, they randomly assign child words to the parent trained embeddings. They showed that their transfer learning improves the performance of NMT systems in translation from Hansa, Turkish, and Uzbek into English with the help of a French→English parent model. Unfortunately, the authors have not experimented with translation direction from English. Shared-target language scenario is an easier task as we show in Section 5.2. Furthermore, they got the biggest improvements when using the transferred model as a re-scorer of SMT approach.

Nguyen and Chiang (2017) build on top of the work of Zoph et al. (2016). In contrast, they focused on a scenario where both parent and the child are low-resource and linguistically related. They improved the transfer learning by using a BPE vocabulary instead of word-level vocabulary as in Zoph et al. (2016). The vocabulary is prepared in advance from both parent and child training data, i.e. they use the warm-start scenario. Their approach is similar to our Balanced Vocabulary with several differences. They used transliteration of Arabic script in order to normalize training corpora to Latin script and used segmentation to increase the number of overlapping tokens between languages. Furthermore, they have not balanced the corpora in any way. With that setup, they show that linguistic relatedness plays an important role and that transfer learning also helps in a scenario where both parent and child are low-resource. They demonstrated that transfer learning works better on subword-level NMT than on word-level mainly due to the higher overlap of tokens between languages. Additionally, they showed that freezing parent parameters such as English embeddings (Zoph et al., 2016) is not necessary for the transfer learning and can even hurt its performance. Unfortunately, they also evaluated only the scenario with shared-target language, leaving a place to investigate the effect in

the more difficult setting with the shared-source language. In Section 4.7.3, we showed that the relatedness of languages is not necessary for successful transfer learning.

Kim et al. (2019) solved the problem with mismatching vocabulary in cold-start approach by applying cross-lingual word embeddings. They trained monolingual word embeddings (Mikolov et al., 2013) for the parent and child source language and then computed the linear mapping between the two embedding spaces. The child model then continues the training from the parent model with modified word embeddings. The linear mapping is applied only to the encoder embeddings. The method assumes the same target language for parent and child model.

Lakew et al. (2018) experiment with a cold-start transfer learning in the multilingual NMT setting. They tackle the problem with vocabulary mismatch by dynamically adapting the vocabulary in a similar way as our "Vocabulary Transformation Technique" (see Section 4.6). During the transfer, the parent vocabulary items that are not used by the child are replaced by random tokens from the child vocabulary. However, in contrast to our technique, they reset embedding weights for the randomly assigned words. They evaluate the method on the multilingual setting of four language pairs, where instead of transferring from one parent to one child, they transfer parameters consecutively from one parent to the first child, followed by transferring from first child to the second child and lastly finishing by transfer learning of the third child. Before each change of training corpus, the vocabulary is dynamically updated for the upcoming language pair. They proposed two approaches: either incrementally add training data to the parent training corpus, thus increasing the number of training sentences with each consecutive parent or to use only the current child's training data. The former approach can prevent worsening the performance of the child on previous parent testsets. However, the latter approach reaches a better score on the latest child language pair with the disadvantage of losing the performance on the earlier parents.

Neubig and Hu (2018) combined the multilingual MT (Johnson et al., 2017) with transfer learning by Zoph et al. (2016). They train a multilingual general system that can translate between all investigated language pairs followed by specializing the model to one high-resource language pair. They trained the multilingual parent on 58 language pairs. Their main goal is low-resource languages for which they come up with the technique of using a helper related language pair, which is mixed into the training set. For example, they mix Slovak and Czech sentences aligned on the English side. They confirm results from Johnson et al. (2017) that using a single-pair baseline, in this case, mixing the related languages, performs better than the multilingual model. However, when they used the transfer learning technique on the multilingual parent model, the performance significantly improved over single-pair baseline, which shows the usefulness of multilingual parent in transfer learning.

4.10 Conclusion

We proved transfer learning to be an effective technique for NMT under low-resource conditions both by our experiments and the related work. Existing

methods require a shared-target language, language relatedness, or specific training tricks and regimes. In contrast, we describe our approach to transfer learning leveraging these constrains in both cold-start and warm-start setting. Furthermore, we showed that the relatedness of languages is not critical for transfer learning, nor we needed transliteration for our methods. Another important ability of our approaches is to handle language pairs that have different target language between a parent and child model.

In this chapter, we have explained the transfer learning technique on NMT and discussed a difference between cold-start and warm-start techniques. We proposed two techniques: Direct Transfer and Vocabulary Transformation techniques in the cold-start setting, where we investigated the influence of parent vocabulary on the child language pair. In Section 4.7, we studied the warm-start scenario and proposed Merged and Balanced Vocabulary techniques. Furthermore, we evaluated these approaches as well as using child-specific vocabulary for the parent training, which surprisingly performed worse than other techniques. Lastly, in Section 4.8, we compared our approaches and showed that Balance vocabulary obtains the best performance out of proposed methods. However, the Transformed Vocabulary needs much less training steps and obtains results similar to Balanced Vocabulary.

Note that all our presented methods are relatively simple to implement with current frameworks and training procedures, which could be a motivation for the community for applying them. We believe that our Vocabulary Transformation technique can replace standard approaches of training models from random initialization as it reaches better performance in a shorter or comparable number of training steps relative to the training from random initialization.

We want to analyze what is behind the improvements, and deeper understand the technique. Therefore, in the next chapter, we focus on an extensive analysis of transfer learning with an ambition to explain how transfer learning works and if we can use this knowledge to improve the transfer learning technique.

5 Analysis

In the previous chapter, we showed the advantages of the transfer learning technique and the gains it brings. However, neural networks are used without us rigorously understanding the exact gains behind individual techniques. Thus neural networks gained the reputation of being a black box.

As Rudin (2018) said, "A black-box model is either a function that is too complicated for any human to comprehend or a function that is proprietary; it is a model that is difficult to troubleshoot. Deep learning models, for instance, tend to be black boxes because they are highly recursive." In other words, neural networks are difficult to understand for their deep recursive architecture and often unpredictable behavior. For example, in image classification, Goodfellow et al. (2014) showed that NN could be deceived by a slightly modified image, which is indistinguishable from the original for humans, yet the NN cannot classify it correctly.

In this chapter, we try to investigate what is really behind the achievements of transfer learning by analyzing the behavior, training process, and what is transferred from the parent to the child. We dive into the analysis of transfer learning.

Although we presented several cold-start and warm-start methods in Chapter 4, we are going to analyze only the warm-start Balanced Vocabulary approach in order to make the analysis consistent. We decided on this approach as it reaches the best performance. However, we are convinced that other presented methods behave similarly.

This chapter is organized into sections that analyze various aspects of transfer learning. We start by investigating the negative effects of transfer learning in Section 5.1. In Section 5.2, we discuss the differences between the scenarios with shared-source and shared-target language. Then in Section 5.3, we try to answer if the training data size is more important than the language relatedness. We follow with a discussion if the gains stem from linguistic features or simply from a better initialization of the neural network in Section 5.4. We conclude

the results from analysis in Section 5.5. Lastly, in Section 5.6, we perform a case study an application of transfer learning on backtranslation.

5.1 Negative Transfer

In generic machine learning, transfer learning is also known for its downsides (Pan and Yang, 2010; Weiss et al., 2016). When transferring knowledge from a less related task, it may hurt the final performance on the child task in comparison to the performance obtained without transfer learning only with the use of a child model. This harmful effect is called "Negative transfer".

The main reason behind the negative transfer is often the domain mismatch between the parent and child tasks or even an unrelated parent domain (Pan et al., 2010; Ge et al., 2014), which prevents the model from the utilization of the parent model during transfer learning.

Wang et al. (2019) proposed a formal definition of the negative transfer and evaluated the definition on several transfer learning approaches. They evaluated the following three critical factors influencing the negative transfer:

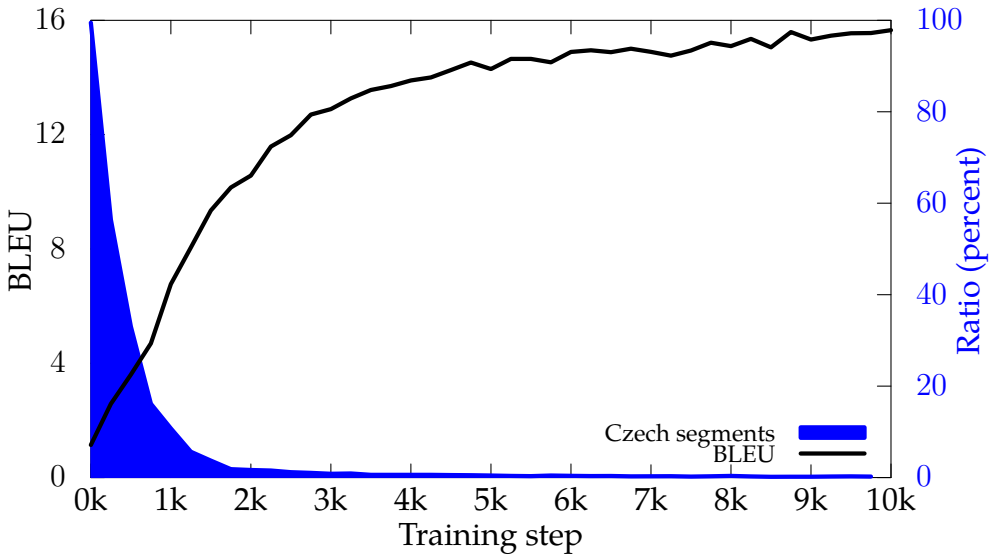
1. Divergence between the joint distributions of both tasks is hurting transfer learning.
2. Effectiveness of transfer learning depends on the size of child data.
3. Transfer learning should be evaluated with the same setting of the neural network to avoid adding a risk of different setups.

As for the first factor, the ideal transfer learning should figure out and take advantage of only the similar parts of tasks, however, in the real-life scenario it often takes into account also the misleading information learned from the parent task. The second factor elaborates that the less training data is available in the child domain, the more fragments are preserved from the parent task, which decreases performance on the child task. On the other hand, when we have plentiful of child data, a better baseline can be trained, which reaches a better performance than transfer learning. Thus negative transfer is relatively more likely to occur. The last factor is to avoid misjudgment by comparing the performance of transfer learning with a baseline using different parameters. For example, fine-tuning hyper-parameters separately for the transferred model and baseline will likely lead to different results due to the hyper-parameter setting.

Transfer learning in the field of NMT emerged recently (Zoph et al., 2016). Thus there is a lack of research on the negative transfer in this field. Zoph et al. (2016) have not discovered any problematic behavior of transfer learning. It could be due to the design of the experiment that avoids the negative transfer altogether. For example, initial works on transfer learning in NMT examine only linguistically related language pairs (Nguyen and Chiang, 2017; Neubig and Hu, 2018). Another possible explanation is that the neural networks are robust enough that they can re-train any transferred parameters hence avoiding the negative transfer at all.

In this section, we try to shed some light on the negative transfer in NMT by evaluating several experiments and identifying the possible downsides of

Figure 5.1: The graph represent behavior of child model during first 10k training steps. The blue area represents the ratio of Czech segments in the output of child model immediately after the transfer learning start. The black curve illustrates the BLEU score on the English→Estonian development set.



transfer learning in NMT. In Section 5.1.1, we investigate if the parent target language leaks to the outputs of the child. Then in Section 5.1.2, we study the condition of having an extremely low-resource child and if that hurts the transfer learning success. In Section 5.1.3, we study the reverse effect of having a parent with less parallel sentences than the child. Wang et al. (2019) mentioned that another factor negatively influencing transfer learning is a divergence between parent and child distributions. Thus we investigate the scenario with no-shared language in Section 5.1.4. Lastly, we conclude in Section 5.1.5.

5.1.1 Traces of Parent Language Pair

Wang et al. (2019) mention as an effect of negative transfer that parent fragments appear in the child outputs whenever the child task has a low amount of data. This is mainly because the child has not entirely forgotten the parent task. In this section, we investigate traces of the parent language pair in child translation, such as text fragments.

During transfer learning, the neural network is not notified about the change in the language pair. This means that the NN has to forget the parent task during the training on child parallel corpus. This can result in fragments of parent target language appearing in the child model output.

In order to test if there are any traces of the parent language pair in the output of the child model, LanideNN (see Section 2.2) automatically identifies the language of transfer learning outputs, and we measure how often does the parent language pair appear in the child output.

We used our LanideNN as it is trained to recognize language switching within one sentence instead of labeling the whole sentence by one label; we can

Table 5.1: Relic words from parent language in the child output. The English gloss is our (manual) translation of a given Czech word.

Appearances	Czech	English Gloss
49	kámo	buddy
31	Článek	Article
20	Podívej	Look
17	jasný	clear/ok
11	Odpověď	Answer
8	strýc	uncle
7	Poznámky	Notes

get distribution over each character. We calculate the score by labeling each character in the testset with the language label and then calculating the ratio of labels for each language in the testset.

We evaluate the model every 250 steps, which is roughly every three minutes of training. As the parent model we use English→Czech and the child model is English→Estonian. These language pairs have different target language from different language family, which should help when automatically recognizing the language. For this evaluation, we use the English→Estonian development set and $T2T_{11}$.

From Figure 5.1, we see that the NMT model quickly forgets generation of Czech sentences, after just 3k steps the model generates less than 1% of Czech data. The training time for 3k steps took only 43 minutes. We remind the reader that the standard time of training even extremely low-resource language pairs is at least 50k training steps. Therefore it takes only a fraction of time for the NN to forget the parent target language.

Observation 17: *During the training of a child model, the NN almost instantly forgets the parent target language and adapts to the child target language.*

We need to mention that the results are based on an automatic measure and that LanideNN's error rate, needs to be taken into account (in a multilingual setting, the error rate is less than 4%, see Section 2.2.3). We note that this automatic language detection cannot be reliably used for fine-grained evaluation to investigate if the child occasionally generates the parent target language.

In order to evaluate how often child model produces parent (Czech) words, we used the final child model to translate 100k English sentences randomly selected from the parent training corpus. We chose the parent training corpus as these sentences could be memorized by NN from the parent model training and it is thus most likely that the training input sentences could trigger the parent behavior in the child model. The final child model was trained for 75k steps.

We evaluate the translated sentences both automatically as well as manually. The automatic evaluation is the same as in the previous analysis and identified 54 Czech sentences. We manually checked these sentences and found out that a few of them are Czech postal addresses or other named entities, the rest are Estonian sentences with Czech words inside them. The longest Czech sequence

without named entities is "u všech čert" (an incomplete idiom "all the hell"). However, we noticed that the Czech words are often already present in the source sentence.[1]

In order to analyze the size of Czech words produced by the child model, we listed all sentences that use Czech characters not contained in Estonian, e.g. characters with diacritics. Then we removed sentences, where the source also contained Czech-only characters. This way, we got 624 out of 100k sentences, which often contained only one word that we manually identified as Czech. Altogether these sentences contained 645 words with Czech-only characters (the evaluated dataset contained 1.4M words). We list few of the most frequently appearing in Table 5.1.

Despite that we evaluated the child model with parent training data, i.e. the corpus that the model could have memorized, we found only a few parent relics. Therefore, we conclude that transfer learning is not negatively affected by the parent model.

Observation 18: *Relic words (i.e. words from the parent target language) are very rare in NMT transfer learning.*

The rapid change in behavior, when only 3000 steps are enough to forget the parent target language, is one of the results of "catastrophic forgetting", a nature of a network to quickly forget or re-train previously learned features. This phenomenon has been widely studied (Kirkpatrick et al., 2017; Kemker et al., 2018) as the researchers develop methods to overcome this issue. Furthermore, we have tackled problems connected with catastrophic forgetting in Kocmi and Bojar (2017a) when experimenting with curriculum learning (Bengio et al., 2009).

Despite catastrophic forgetting being a problem in machine learning in general, in the scenario of transfer learning, we believe it helps to avoid negative transfer from the parent model by forgetting it. However, we need to keep in mind that in future, when algorithms become more robust in terms of catastrophic forgetting, the negative transfer could emerge as a problem for transfer learning because we saw some words attributed to the parent language pair.

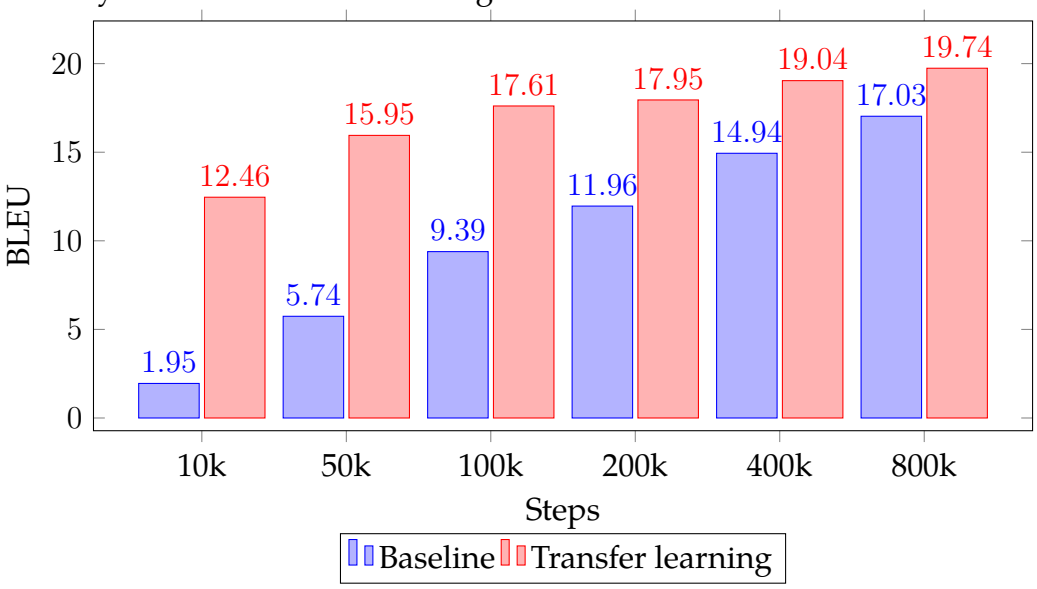
5.1.2 Extremely Low-Resources Language Pairs

We showed that NMT systems quickly forget the parent language in the low-resource scenario. Now, we evaluate an *extremely* low-resource language pair to find out, if our approach helps in the extremely low-resource scenario (see Section 2.1.2) or if insufficient data lead to negative transfer as Wang et al. (2019) described. The results in this subsection are from our paper Kocmi and Bojar (2018) ($T2T_4$).

We simulate extremely low-resource settings by downscaling the data for the child model but maintaining the same parent model. It is a common knowledge that gains from transfer learning are more pronounced for child models with smaller training data. We use the English→Finnish as a parent model for English→Estonian. We mention that shared-source is the harder transfer scenario as the model cannot benefit from the parent English language model

[1] The Czech–English training corpus is noisy, as we discussed in Section 2.3.1.

Figure 5.2: Maximal score reached by English→Estonian child for decreasing sizes of child training data, trained off an English→Finnish parent. The baselines use only the reduced Estonian–English data.



because the target language changes from Finnish to Estonian (more analysis in Section 5.2).

The results of downscaling the child training corpus are shown in Figure 5.2. We see that our approach applies even to extremely low-resource language pairs with as few as 10k sentence pairs. We see this behavior on the 10k training corpus, where the baseline reaches 1.95 BLEU. This behavior is in accordance with observations done by Koehn and Knowles (2017). For such a small amount of training data, the NMT baseline cannot be properly trained. With transfer learning, NMT suddenly becomes able to train the model and reaches 12.46 BLEU.

Observation 19: *Transfer learning helps NMT to train models for extremely low-resource language pairs that are not possible to properly train on their own.*

Sennrich and Zhang (2019) recently revisited the problem of extremely low-resource language pairs and showed that it could be tackled with various tricks. Furthermore, transfer learning could be used as another way of improving the extremely low-resource language pairs hand in hand with other techniques mentioned by Sennrich and Zhang (2019).

As Wang et al. (2019) summarized, transfer learning can lower the performance of the child task whenever the amount of child training data is too low. We showed that this is not the case in NMT because transfer learning can help training the model even when the baseline cannot be trained in the first place.

5.1.3 Low-Resource Parent Language Pair

We showed that transfer learning is not restricted to the low-resource scenarios and improves the performance even when the child is a high-resource language pair (see Observation 9 and Observation 16). However, it is in the scenario, where both the parent and child model are high-resource.

Table 5.2: The column "Transfer" is our warm-start method, baselines correspond to training on child corpus only. We show the sizes of corpora in millions sentences. The ‡ represents significantly better results.

Parent	Size	Child	Size	Transfer	Baseline
EN→Estonian	0.8M	EN→Finnish	2.8M	**20.07** ‡	19.50
Estonian→EN	0.8M	Finnish→EN	2.8M	23.95	**24.40**
EN→Slovak	4.3M	EN→Czech	40.1M	22.99	**23.48** ‡
Slovak→EN	4.3M	Czech→EN	40.1M	28.20	**29.61** ‡

Wang et al. (2019) summarized that transfer learning is negatively influenced by the parent model whenever the parent has a low number of training examples. In this section, we examine this condition in the area of NMT, and we investigate the scenario where the parent has a lower amount of parallel sentences than the child model.

Zoph et al. (2016) conclude that the relatedness of languages is the main factor influencing the success of transfer learning, which we already showed it is not a necessary condition in Chapter 4. However, we use linguistically related languages in this experiment, because the secondary goal is to test what plays a bigger role in transfer learning – the relatedness of languages or the size of parent data.

Results from Kocmi and Bojar (2018) ($T2T_4$) are presented in Table 5.2. We see that low-resource parents do not generally improve the performance of sufficiently resourced children. The only exception is the child English→Finnish, where the child has only 3.5 times more parallel sentences than the English→Estonian parent.

Observation 20: *Transfer learning harms the child performance whenever the parent has substantially less training data than the child.*

Whenever the child has notably more training data, e.g. ten times more for Czech–English it even (significantly) decreases the child's performance compared to the baseline. Therefore we conclude that transfer learning is negatively influenced in scenarios where the parent has substantially less training data. We suppose it could be due to the initial warm-up steps when the network changes rapidly, thus low-resource language can skew it. However, more analysis is needed to study this behavior properly.

Furthermore, we evaluated linguistically related languages where the relatedness could help improve the model. For example the Czech and Slovak are related to such extent that evaluating English→Czech system on English→Slovak testset output leads to 6.51 BLEU (see Section 4.7.3). However, the relatedness did seem not to play any role in our experiments, and transfer learning led to worse performance than training on child parallel corpus only.

Observation 21: *For a high-resource child the linguistic relatedness of parent and child language pairs is less important than the size of the parent training corpus.*

Table 5.3: No-shared language scenario of transfer learning. The child model is Estonian→English. Each row represents various metrics for measuring MT performance, where higher number is better for all metrics. The significance ‡ is computed pairwise relative to the baseline "No transfer learning".

Parent	BLEU	nPER	nTER	nWER	chrF3	nCharacTER
No transfer learning	21.74	54.33	35.66	32.70	49.87	37.70
Arabic→Russian	22.23	55.05	36.66	33.59	50.86	**40.11**
Spanish→French	22.24 ‡	**55.32**	36.58	33.69	50.88	39.59
Spanish→Russian	**22.52** ‡	55.26	**36.85**	**33.79**	**51.28**	39.92
French→Russian	22.40 ‡	54.99	36.50	33.39	50.93	39.60

5.1.4 No-Shared Language Scenario

One of the main factors of negative transfer is the divergence in distributions between parent and child training data (Wang et al., 2019). In NMT, one would assume that the languages in question are the key element affecting task similarity.

In Section 4.7.3, we showed that the relatedness of the languages is not the most critical for transfer learning. Moreover, the related English→Finnish parent performed worse even when compared to a parent that uses a different writing script, in our case, English→Russian with Cyrillic. Thus we have not detected any negative transfer when evaluated on less linguistically related languages.

However, our experiments always contained a language shared between the parent and child, e.g. English, which could work as a connecting bridge during transfer learning, thus preventing the negative effects. In order to test the negative transfer in NMT, we experiment with a no-shared language scenario.

We examine the performance of Estonian→English child trained on top of parents using unrelated languages, specifically Arabic→Russian, Spanish→French, Spanish→Russian, and French→Russian. The parents are trained with the UN corpus (Ziemski et al., 2016), which has 10M multi-parallel sentences across six languages.

The results from Kocmi and Bojar (2018) ($T2T_4$) are shown in Table 5.3. We see mostly significant gains from transfer learning in all cases. The only non-significant gain is from Arabic→Russian, which does not share the script with the child's Latin at all, only sharing of punctuation and numbers is possible across all the tested scripts.

Observation 22: *Transfer learning improves the performance even for the no-shared language scenario.*

There is no loss of performance in comparison to the baseline. This can be seen either as the evidence that transfer learning is not negatively affected by the difference in data distributions between parent and child, or that the mere distributional properties of all (tested) languages are sufficiently similar to be useful for transfer learning in NMT.

Surprisingly, the Spanish→Russian (with a target languages that uses the Cyrillic script) reached a better performance than the Spanish→French, a target language that is linguistically closest to the Estonian→English from all four investigated language pairs. However, neither of these two systems is significantly better than the other. Furthermore, the gains are quite similar (+0.49 up to +0.78 BLEU), which supports an assumption that the major factor influencing transfer learning is the size of the parent (here, all parents have 10M sentence pairs). We are going to discuss this aspect in Table 5.5. This result can also be explained with a similar domain of parent training set. In comparison, the Czech→English parent, which has 40.1M sentences from a broader range of domains and has a shared language (English), improved the performance of Estonian→English by 3.38 BLEU.

Observation 23: *The exact parent language pair does not seem to affect the performance given a particular domain and parent data size.*

In the no-shared language scenario, the gains cannot be attributed to the language model or model parts such as shared English word embeddings. The subword vocabulary overlap is mostly due to short subwords or numbers and punctuation.

In Section 2.4.2, we discuss issues with the BLEU metrics, e.g. ignoring the importance of various n-grams or high-influence of tokenization. For this reason, we computed the scores for several other automatic methods. We see that in all metrics, transferred models perform better than the baseline.

The experiments presented in this section indicate that the parent simply works as a better model weight initialization in comparison to the random initialization. We investigate it more in Section 5.4.

5.1.5 Conclusion on Negative Transfer

In this section, we investigated the effect of negative learning. We evaluated various scenarios and identified two weaknesses of transfer learning. We showed that the final child model could exhibit relics of the parent target language in Section 5.1.1. Moreover, in Section 5.1.3, we showed that when the parent has substantially less training data than the child, transfer learning may hurt the final performance. In this scenario, the training model without transfer learning performs better.

Lastly, we reevaluate three critical factors proposed by Wang et al. (2019) that influence the negative transfer:

1. Bigger divergence between the distributions of both tasks is hurting transfer learning.
2. Effectiveness of transfer learning depends on the size of child data.
3. Negative transfer should be evaluated with the same setting of the neural network, to avoid adding risk from a different network.

We showed that divergence in distribution does not play a negative role in NMT transfer learning because even the no-shared language scenario improves child performance (see Section 5.1.4).

Table 5.4: Number of steps needed for a model to converge. The size shows the number of sentences in the corpora of each language. For both child the second language is English. The results are from Section 4.8 with subtracted time of parent model training.

Child language	Size	Shared-source	Shared-target
Estonian	0.8M	75k	25k
Gujarati	0.2M	195k	180k
Russian	12.6M	935k	790k

The second factor plays a role only in the case when the parent has less training data in comparison to the child (see Section 5.1.3).

We used an identical parameter setting for both baseline and transfer learning. Thus the third factor does not play a role in our evaluation.

In conclusion, we provided an analysis of the negative effects of transfer learning and discovered two possible issues of transfer learning. We are not aware of any previous study discussing negative transfer in NMT transfer learning.

5.2 Does Position of Shared Language Influence Transfer Learning?

We noticed that transfer learning has different behavior for the shared-source and shared-target language scenarios. For example, English→Russian parent improved the Estonian child more than English→Finnish (20.41 vs. 19.74 BLEU), however in the opposite direction Russian→English worked as a worse parent than Finnish→English (23.54 vs. 24.18 BLEU), as we showed in Section 4.7.3.

In this section, we investigate the influence of the position of the shared language on transfer learning. Moreover, we discuss which of those tasks is harder for the NMT transfer learning.

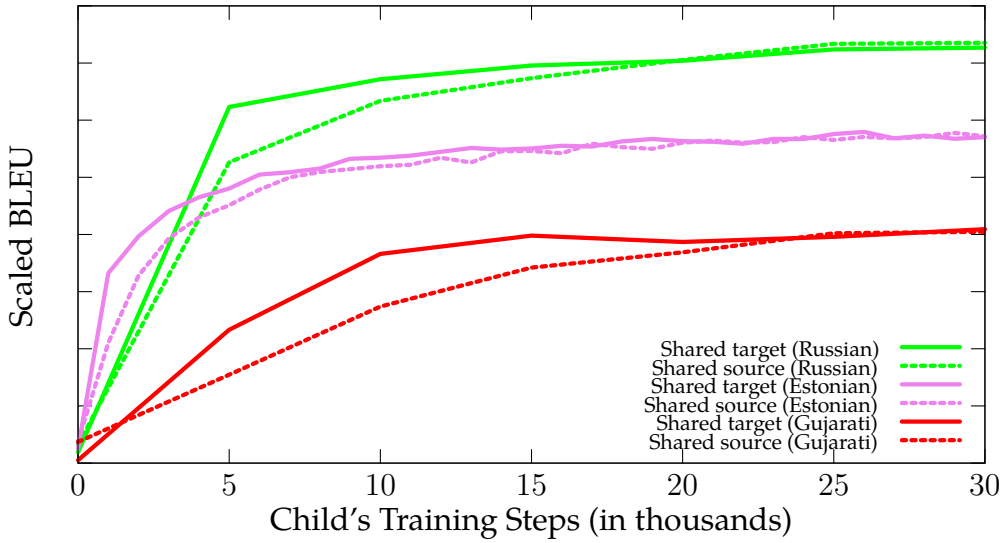
5.2.1 Shared Language Position Effect on Convergence Speed

We start with an investigation of the training time needed for each direction to converge. We recall the results from Section 4.8 ($T2T_8$) in Table 5.4. We also added the Gujarati–English pair from our paper Kocmi and Bojar (2019b) ($T2T_{11}$). We subtracted the number of parent steps needed for the convergence and showed the results of the child model training in Table 5.4.[2]

In Table 5.4 we can see that the shared-target language scenario converges faster for both the low-resource Estonian and the high-resource Russian. In the case of Gujarati–English, the convergence is only slightly faster for the shared-target.

[2]Training steps in Section 4.8 are the sum of parent plus child training steps, but we show only the child number of steps in Table 5.4.

Figure 5.3: Learning curves for various language pairs in both directions. The Y-axis has been scaled for each learning curve by a constant in order to match their final performance. The bracket specifies the child's second language that is paired with English. The convergence is seen only on Gujarati pair as other languages converged later than within first 30k steps.



Observation 24: *Transfer learning with a shared-target language converges in fewer steps than with a shared-source language.*

We need to mention that the total number of training steps does not always reflect the convergence speed because the performance usually fluctuates and thus the model can converge after a different number of steps, which depends on randomness in training. Moreover, the training time does not explicitly mean if the task is easier; there are many other factors like shared language, the noisiness of training data, and other factors.

Therefore we investigate the learning curves and look for a distinctive behavior between these tasks.

5.2.2 Shared Language Position Affects Slope of Learning Curve

In Figure 4.1, we described three impacts of transfer learning on the learning curve – namely, higher start, higher performance, and higher slope. The effect of higher slope suggests that the model trains faster, therefore we compare learning curves of shared-source and shared-target scenarios in order to find out which one learns faster.

We report the learning curve's Y-axis in BLEU, but any other metric could be used. The BLEU score has a disadvantage that it cannot be compared across various languages or even testsets. Therefore, in order to study the slope of the learning curve, we need to scale it. We scale each learning curves by multiplicating the performance (BLEU) by a fixed constant. The constant is selected in order to align the best-reached performance of both translation directions.

Figure 5.3 presents the learning curves of three language pairs evaluated in the previous section on their respective development sets. We investigate only the first 30k training steps, where the difference in slopes is most visible.

When comparing the learning curves of shared-source and shared-target scenarios, we see that for all three language pairs, the shared-target has a higher slope than the shared-source.

Observation 25: *Transfer learning with a shared-target language has a higher slope of the learning curve.*

This observation suggests that shared-target, i.e. having shared language (e.g. English) on the target side, is easier for transferring knowledge from parent to child. In this scenario, NN reaches higher performance in a shorter time compared to the shared-source scenario. This behavior is not surprising. From the neural network's point of view, it is learning to predict the shared language through the whole training process. Therefore it can utilize the language model from a parent with only learning different encoder's part of the model. We further analyze the behavior by freezing various parts of a model in Section 5.4.

5.2.3 Parent Performance Drop

In transfer learning, we do not pay attention to the final parent's performance as is customary in multi-task learning. In Section 5.1.1, we showed that for translation direction with shared-source, the child model quickly forgets the parent target language.

In this section, we investigate how the parent translation deteriorates during the child's training phase in both directions, and if the shared-source and shared-target scenarios behave differently.

We evaluate two scenarios, the shared-source and shared-target under our English→Gujarati and Gujarati→English child models transferred from Czech–English parent (Kocmi and Bojar, 2019b) ($T2T_{11}$).

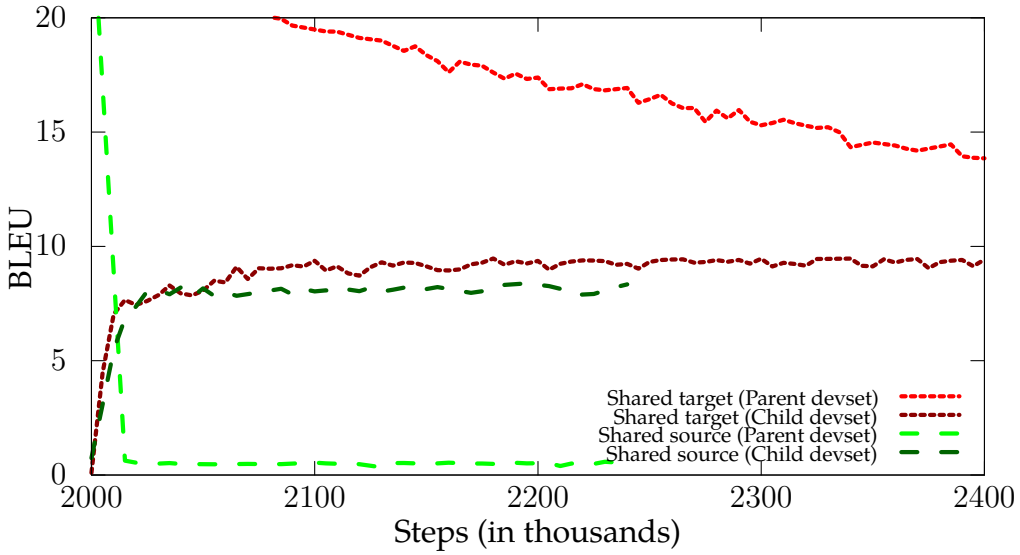
We evaluate both models each by the corresponding child's and parent's development set. The scores between language pairs are not comparable as they are for different languages.

Figure 5.4 presents the results of our experiment. The figure starts at a global step of 2M, the end of the parent models. At this point, all parent models have the final performance higher than 20 BLEU (not visible in the figure). The learning curves with the same dashing correspond to the same model evaluated either on the child or parent development set.

We can see that from the start of the child model's training, the performance of the parent largely deteriorates on the parent's development set and that the behavior for shared-source and the shared-target is distinct. In the scenario with shared-source language, the model does not know to which language it should translate. Therefore it learns to always translate to the child target language.

Interestingly, in the shared-source language scenario, the performance drops nearly immediately. In contrast, whenever we investigate the shared-target scenario, the performance is deteriorating slower, and even after finishing the child's training, it is still able to translate Czech→English parent language pair with 15 BLEU accuracy.

Figure 5.4: Performance of child model on a parent development set. Both child are Gujarati–English. Learning curves are evaluated on a development set of analogous language pairs.



Observation 26: *Parent performance deteriorates during the child training at different speed depending on shared language position. The shared-source language scenario declines almost instantly. The shared-target language scenario deteriorates slowly.*

Figure 5.4 shows that the neural network forgets the parent source language slower. Thus the phenomenon of critical forgetting is mostly concerned with the decoder part. If it forgot both of them in the same way, the drop in parent performance would be similar in both directions, i.e. English→XX and XX→English.

Observation 27: *A converged child model in the shared-target scenario can still translate the parent language pair to some extent. It is not possible in the shared-source scenario.*

This behavior could be a result of an error backpropagation, where the gradient vanishes as it travels back through the network, thus updates the encoder layers less than the decoder layers. This is especially true in transfer learning with shared-target because the network already knows how to generate the target language, e.g. English. Thus it does not produce sufficient errors that would modify the encoder. Thus the encoder does not forget the parent task as quickly. This suggests that the most important part of the model is the decoder and therefore, transfer learning with the shared-target language is easier to learn as it already knows how target language should look like.

As future work, we could increase the error backpropagation to the encoder using ultra-residual connections similar to Emelin et al. (2019) that would connect various layers of the encoder directly with the decoder's layers. Moreover, we could add a special tag at the beginning of the source language specifying the desired target language (Johnson et al., 2017).

5.2.4 Conclusion

In this section, we investigated if there is a difference between the shared-source and shared-target scenario. In Section 5.2.1, we showed that shared-target converges faster than shared-source, which is associated with a higher slope as demonstrated in Section 5.2.2. Furthermore, we revealed that NMT forgets the parent model slower in the shared-target scenario in Section 5.2.3.

We cannot compare our findings with other works in the NMT transfer learning because we are the first to show transfer learning with a shared-target language. However, we can investigate a closely related task of multi-task NMT. Firat et al. (2016) trained a multilingual system with six languages and showed that their model outperforms the single-pair baseline only in case of the shared-target language. In the case of the shared-source language scenario, the baseline performs on-par or better than their multilingual system. Johnson et al. (2017) showed that the many-to-one scenario (shared-target) leads to improvements in most cases, unlike the one-to-many scenario (shared-source).

In summary, based on previous observations, we conclude that the shared-target language scenario is a harder task for transfer learning compared to the shared-source language scenario.

Lastly, there are other scenarios that do not fall into our investigated categories. We can have the shared language on a different translation side, e.g. parent of English→XX and child XX→English, which we are going to investigate in Section 5.4.2. And we already examined a no-shared language scenario in Section 5.1.4.

5.3 Language Relatedness versus Data Size

Whenever humans learn a new language, it is much easier for them if they know a related language. Therefore we suppose that similar works for NN with transfer learning. Zoph et al. (2016) in their transfer learning approach concluded that "the choice of parent model can have a strong impact on transfer models, and choosing better [related] parents for our low-resource languages could improve the final results". Furthermore, the use of related language pair as a way to improve the performance of a model has been widely studied, and researchers showed that related language pairs can be used as a source of improvements (Nakov and Ng, 2009; Nguyen and Chiang, 2017).

However, in Section 4.7.3, we saw that English→Russian is a better parent to English→Estonian than English→Finnish despite being linguistically related, moreover having the same script.[3] Thus the main difference could be in the number of parent parallel sentences, where Russian–English has 12.6M sentences, and Finnish–English has only 2.8M sentences.

Besides, we showed in Section 5.1.4 that entirely unrelated languages still yields improvements in the child model. On the other side, having less resourceful parent can harm the performance of the child as we showed in Section 5.1.3.

These are indications that the relatedness of languages is not the main factor in transfer learning, and the size of the parent model has a bigger influence on

[3] We do not transliterate the Cyrillic as done by other works (Nguyen and Chiang, 2017).

Figure 5.5: Various ways of damaging original sentences. Each column corresponds to the original word. All occurencess of a word type are replaced with the same string anywhere in the corpus, except for option 1 with shuffled words.

Original:	my	cat	likes	playing	with	my	other	cats
Option 1:	likes	playing	other	cats	my	cat	with	my
Option 2:	has	juggling	rather	study	research	has	those	set
Option 3:	tf	jha	sprlz	wshfpun	dpao	tf	vaoly	jhaz

child performance. In this section, we investigate the phenomenon of language relatedness in contrast to the training size of the parent model.

5.3.1 Artificially Related Language Pair

Many factors influence language relatedness: linguistic family, writing script, grammar phenomena, etc. Therefore it is hard to measure the relatedness of various languages, especially comparing training data sizes with various degree of relatedness. In this section, we evaluate the effect of various degree of relatedness in contrast to various training data sizes. We present artificially related language where we can influence the degree of the language relatedness and measure the performance of the child. The artificial language is prepared by harmful modifications of the original training set.

There are several options on how to prepare artificial related language pair from the original training data. We can either shuffle words in the sentence, which creates a language pair with the same vocabulary, but different word order. The second option is to shuffle words within the language, e.g. "cat" would always be replaced by "juggling". The third option is to shuffle individual characters within each word type based on an exact replacement rule. The variants are visualized in Figure 5.5.

The first option is prone to the word order, if we would like to have artificial language with consistent word order we need to rely on some linguistic analysis, which would add another layer of uncertainty to this experiment, however, we investigate this option in Section 5.4.3 as a way to study word order). The second option is problematic as we want the related language to have a property that similar words behave similarly, for example, a word "cats" should be replaced with something similar to "juggling", however this would need an in-depth analysis of clusters of words and generation of rules which words are mapped to which. It is very hard, especially for an inflected language such as Czech. The third option generates a language with the same word order and language where visually similar words appear in a similar context. However, the language is unintelligible from the original. Moreover, due to the subword segmentation, the actual lengths of sentences as seen by NMT vary and thus the NN cannot learn easy mapping across the sentences.

We decided to create the artificial language pair by the last option of substituting characters. We mix only the alphabet characters to match real-life conditions where most languages use the same punctuation and numbers. Furthermore,

Figure 5.6: An example sentence in various ratio of substitution.

Original:	**Pardon? Have you seen this cat?**
70% related:	**Pardon?** Crnk **you seen this** tre?
50% related:	Irfjwh? **Have you** ykkh ecsy **cat**?
30% related:	Irfjwh? Crnk bwm **seen** ecsy **cat**?
0% related:	Irfjwh? Crnk bwm ykkh ecsy tre?

we preserved the capitalization. We modify both source and target language. Therefore there is no shared unmodified language between parent and child.

In order to scale the relatedness of languages, we randomly select X% of words from source and target language and substitute characters only in the remaining words; thus we obtain corpus with X% words unchanged. This way we get a pseudo related language pair with a varying degree of relatedness where 0% is almost unrelated, and 100% is identical language. An example is in Figure 5.6.

In this experiment ($T2T_{11}$), we use English*→Czech* as a parent model with various degree of relatedness and various amount of training data. The corpus is created from CzEng 1.7 (see Section 2.3.1). We experiment with 80%, 50%, and 0% related corpus and each in 2M, 5M, 10M, and 20M parallel sentences. We use a warm-start technique where all models use the same vocabulary that is created from 50% related corpus.

As the child language pair, we use 100k unmodified random sentences from the same corpus that have not appeared in our parent corpus. The performance of a child when trained solely on its training data is 7.17 BLEU.

The training process is as follows: train the parent model for 1M training steps, take the last model, and continue with the training of the child model for additional 200k steps. The best child model is selected based on the development data and evaluated against English→Czech testset.

The results in Table 5.5 present an interesting pattern. We can see that having more data can be more useful than a related language with fewer data. For example, 50% parent with 2M parallel sentences reached 16.76 BLEU, in contrast, having ten times more data but 0% related parent yields 18.25 BLEU. On the opposite, whenever the difference is only double, then the relatedness helps more with fewer data (19.13 BLEU vs. 18.25 BLEU). With 5M training data, the 50% parent already performs better than 20M with 0% (19.63 BLEU vs. 18.25).

We obtained similar results in real-life experiments as noted in the previous chapter, where more resourceful language (Czech or Russian) performed better than related language with less data (Finnish) with the Estonian child.

Observation 28: *Language relatedness plays a role in transfer learning. However, the amount of parent training data can improve performance even more than language relatedness.*

This finding can help when deciding which parent language pair to choose for a particular child. For example, whenever related language pair does not have enough training data, we could choose any training corpus with a high number

Table 5.5: The results of various parent models. Each column specifies the size of parent training data, which is randomly downsampled from the original. Each row specifies parent model relatedness. The scores are in BLEU and specify performance of child model trained from the parent. For models with the star the performance dropped quickly during child training.

	2M	5M	10M	20M
80% related	18.11	*20.46	*21.81	*22.10
50% related	16.76	19.63	19.13	19.61
0% related	15.09	16.83	17.91	18.25

of parallel sentences (for example Czech–English). However, our experiment is performed on artificially related languages, which could be considered as a noisy parent model. Moreover, we evaluated only 12 settings of relatedness and training size.

We noticed that with 80% relatedness the child model (labeled by a star) performs best without the training on its training data as the performance quickly deteriorated during the child training.

The language relatedness is not the only criterion or the most important one. Even unrelated parent can improve the performance of the child. In the next section, we examine the effect using parent language pair with an artificially huge number of parallel sentences.

5.3.2 Parent Trained on Large Mix of Languages

We showed that even completely unrelated language can yield improvements in the child model and that the number of parallel sentences plays an important role. Therefore we investigate scenario, where we create the biggest possible training corpus by mixing all corpora we have at hand.

We created artificial training corpus that translates between English and a mix of languages. Our target child is Estonian–English. Therefore we avoided the training corpus of this language pair. We collected training corpora of most languages from WMT 2019, most of them we already used throughout this thesis. Then we mix them together by aligning the English on one translation side and putting the other languages together on the other side. This way, we create Mix–English corpus.

We have not added any target language label as is used in the multi-task learning (Lu et al., 2018) so our resulting model is not suitable for any practical translation; it picks a target language at random for every input segment.

We collected following language pairs with various training sizes: German-EN with 42.2M sentences, Czech-EN with 40.1M, Russian-EN with 14.7M, Italian-EN with 8.3M, Slovak-EN with 4.3M, Finnish-EN with 2.8M, Dutch-EN with 2.6M and Basque-EN with 0.6M parallel sentences. The final shuffled corpus has 117M parallel sentences.

Table 5.6: The result of Estonian–English child trained off of various parent models. The ‡ represents significantly better results.

	Czech (Warm-start)	Mix (Cold-start)
EN→Estonian	20.07	**20.72** ‡
Estonian→EN	23.35	**24.76** ‡

We trained two models ($T2T_{11}$): English→Mix and Mix→English for 11.6 million training steps each. For training, we have used multi-GPU training with 8 NVidia GeForce 1080 graphical cards.

We decided to use our cold-start technique in order to leave the vocabulary selection for the child instead of warm-start transfer learning, which specifies a vocabulary in advance of parent training.

Before we evaluate transfer learning, we investigate the trained parent models. We took the Estonian–English testset and translated it with both English→Mix and Mix→English models. We have noticed that the English→Mix generates outputs in various languages but never mixes them within a sentence. For example, whenever the translation starts with Czech, the entire output is in proper Czech. Thus we automatically evaluated the translated sentences and counted how many languages appear in the output of translated Estonian–English testset:

- Mix→English – 80.4% Estonian, 18.2% English and 1.4% others.
- English→Mix – 43.1% German, 26.5% Czech, 22.0% Russian, 3.5% Italian, 1.7% Dutch and 3.2% others.

Mix→English often only copies the source into the target. Therefore it generates mostly Estonian sentences (sentences copied from the source). English→Mix on the other hand somewhat randomly select translation language, we can see it reflects proportions from training corpus, which contains the most sentences from German and Czech, followed by the Russian and Italian.

Observation 29: *NMT reflects the domain (language) distribution of sentences from the training corpus despite any additional knowledge.*

Results of Estonian–English child trained off of Mix–English parent language pair are in Table 5.6. Before discussing the result, we have to mention that we are comparing the cold-start direct transfer (column "Mix") with warm-start balanced vocabulary (column "Czech"). The reason is that we trained the Mix model by $T2T_{11}$ and we realized too late that our cold-start experiments are trained by $T2T_8$ setup (see the differences in Section 3.4). Thus instead of training the model again, we are comparing the Mix model with warm-start setup also trained on $T2T_{11}$.

Therefore we remind the results from Section 4.8, where we found out that the warm-start technique always has better performance than the cold-start direct transfer.

In Table 5.6 we see that Mix model performs significantly better than the warm-start technique on Czech. Moreover, based on our findings in Section 4.8 whenever the warm-start technique would train the Mix model, it should obtain even better performance.

This result is unusual on its own because the parent Mix model is not a regular MT system as it was trained on a mix of languages and cannot be used for translating in practice because it generates sentences from various training languages at random.

Observation 30: *More data in the parent model yields better performance of a child even when the parent model is trained on a mix of languages.*

We showed that the parent model does not have to be a model trained on the same distribution of data, for example, the same language pair or similar modification of parent training data as in Section 5.3.1.

5.3.3 Conclusion on Language Relatedness

We illustrated that less related language pairs with a higher number of parallel sentences can improve the performance of the child more than a related language with less parallel sentences. For example in Section 4.7.3 in Table 4.10 we showed that English→Czech parent works better for English→Estonian than linguistically related English→Finnish (20.41 BLEU vs. 19.74 BLEU). Furthermore, even languages with a different script and more training data such as English→Russian can improve the performance of the child English→Estonian more than the related English→Finnish (20.09 BLEU vs. 19.74 BLEU). On the other hand, whenever the shared language (in our case English) is on the target side, the more related language Finnish→English is a better parent to Estonian→English than Russian→English (24.18 BLEU vs. 23.54 BLEU). These examples show that the relatedness of the parent is important. However, the size of parent corpus also plays a critical role in transfer learning.

In Section 5.1.4 we experimented with languages that have both parent languages different from the child. Moreover, they have the same number of parallel sentences (10M sentences). We observed that the improvements of the child model have been similar across all four parents, which suggest that the same amount of training data leads to the same improvements. The improvements were even for languages that do not share writing script.

In Section 5.3.1 we showed on an artificial parent that language relatedness plays a role whenever the differences in training data are small (e.g. twice as much data), however having parent trained on much more parallel sentences leads to better child performance regardless the relatedness.

Section 5.3.2 we confirmed the observations by training a parent model on a mix of all training corpora we have available and obtaining improvements in child performance.

Based on our observations across the thesis, we conclude that although the relatedness of languages plays a role the size of training corpus seems to overcome the relatedness whenever the parent can be trained on much more parallel sentences, even up to the point where the parent model is not useful to the translation by itself.

5.4 Linguistic Features versus Better Initialization

Neural networks have the reputation of being a black box. In this section, we try to understand if the gains can be attributed to some linguistic features or if the main contribution of transfer learning is simply a better initialization of weights than random initialization. There are several possible explanations of what are the reasons for the improvements:

- Knowledge of the shared language, e.g. English.
- General linguistic knowledge transferred from the parent model (e.g. word order patterns or typical lengths of sentences).
- Better hyper-parameters that are changed after the start (especially learning rate).
- Better initialization of weights (weights transformed from the parent could have more suitable distributions than the random initialization).

The main contribution could be due to the shared language between parent and child, e.g. English. Although it is possibly one of the main aspects, it is not the only one. In Section 5.1.4, we showed that transfer learning also helps languages that have both source and target language different from the parent model, i.e. Spanish→Russian helping Estonian→English.

Zoph et al. (2016) showed that transfer learning does not utilize only the shared English, but also other parameters from the second language. Thus other options could lead to the improvements of the child model.

Alternatively, the improvements could yield from linguistic features. For example, the child could transfer some knowledge about the word order or the ratio in source and target lengths. On the other hand, the main benefits could be attributed only to NN layout. If we compare the training model from a random initialization or a parent model, there are two main differences. The first difference is in the initial weights, where transfer learning has weights initialized already in some informed part of the parameter space compared to the random initialization. The second difference is in the learning rate because we are using non-constant learning rate depending on the global number of steps it changes through the training process, which can simply mean that different learning rate could lead to the same improvements.

In this section, we start by discussing the effect of shared language on the NMT by analyzing various layers of the network in Section 5.4.1 as well as investigating the behavior of the model when the shared language changes position from parent to child in Section 5.4.2. Then we explore some linguistic features of parent training data and analyze the generated output in Section 5.4.3, Section 5.4.4 and Section 5.4.5. We conclude the section by analysis of learning rate influence in Section 5.4.6 and comparing transfer learning to random initialization in Section 5.4.7.

5.4.1 Freezing Parameters

Thompson et al. (2018) investigated, which parts of a model are responsible for the gains during domain adaptation. They used the technique of freezing

Table 5.7: Child BLEU scores when trained with some parameters frozen. Each row represents a parameter set that was fixed at the pre-trained values of the Czech→English parent. Best result for frozen parts in each column in bold.

Frozen part	EN→Estonian	Estonian→EN
All	1.99	1.39
Embeddings	19.79	22.89
Encoder	19.65	20.61
Decoder	18.76	**23.95** ‡
Attention	19.73	23.00
None	**20.07**	23.35

model sub-networks to gain an insight into NMT system behavior during the continued training.

They segmented the RNN model (Bahdanau et al., 2014) into five sub-networks: source embeddings, target embeddings, encoder, decoder with attention mechanism and the softmax layer responsible for the generation of the output. Then they follow standard scenario of domain adaptation (see Section 4.2).

In this section, we use their technique to evaluate which parts of the neural network are crucial for transfer learning. In order to analyze transferred parameters that are the most helpful for the child model and which need to be updated the most, we follow the strategy by Thompson et al. (2018). We carry out the analysis on Estonian–English pair with Czech–English parent.

Based on the internal layout of Transformer model parameters in the T2T, we divided the model into four parts. (i) Word embeddings map each subword unit to a dense vector representation. The same embeddings are shared between the encoder and decoder. (ii) The encoder part includes all the six feed-forward layers converting input sequence to the deeper representation. (iii) The decoder part is again six feed-forward layers preparing the choice of the next output subword unit. (iv) The multi-head attention is used throughout encoding as well as decoding, as self-attention layers interleaved with the feed-forward layers (see Section 3.3.1). We do not separate the self-attention layers used in the encoder or decoder, therefore when freezing encoder (resp. decoder) we also freeze some of the attention layer matrices.

We run two sets of experiments ($T2T_{11}$): either freeze only one out of the four parts and leave updating the rest of the model or freeze everything except for the examined part.

The results are in Table 5.7. Based on the results the most important part of NN that has to be changed is the decoder for EN→Estonian (resp. encoder for Estonian→EN) that handles the language that changes from the parent to child. With this part fixed, the performance drops the most.

The same observation is confirmed in Table 5.8: all the model parts (including the multi-head attention) can be reused precisely from the parent model as long

Table 5.8: Child BLEU scores when trained with most parameters frozen. Each row represents a parameter set that was free to train; all other parameters were fixed at their pre-trained values. Best result for non-frozen parts in each column in bold. The results marked with * diverged as the model could not train anything.

Non-frozen part	EN→Estonian	Estonian→EN
All	**20.07** ‡	**23.35** ‡
Embeddings	*	*
Encoder	*	13.21
Decoder	7.87	5.76
Attention	6.19	10.69
None	1.99	1.39

as the decoder for EN→Estonian (resp. encoder for Estonian→EN) can learn the new language.

We got a significantly ‡ better score when the decoder was frozen compared to when all the network were free to train in Estonian→English (23.95 vs. 23.25 BLEU). This shows that at least for examined language pair, the Transformer model lends itself very well to decoder reuse. However, we do not see the same in the opposite direction, which confirms that the position of the shared language makes the task different as we discussed in Section 5.2.

Observation 31: *Freezing decoder when the target language is shared during child training can significantly improve the final performance.*

Zoph and Le (2016) needed freezing embeddings for their transfer learning to successfully work. On the other hand, freezing embeddings is harmful to our transfer learning.

Other results in Table 5.7 reveal that the architecture can compensate for some of the training deficiencies. Freezing the encoder (resp. decoder for Estonian→EN) or attention is not that critical as a frozen decoder (resp. encoder).

Observation 32: *Transformer model is robust enough to compensate for some frozen parts and reach a comparable performance.*

Whenever we freeze everything except a particular layer, we get a completely new picture. Results in Table 5.8 show that the network struggles to change the behavior from the parent when most of the network is frozen. Especially the parent embeddings are the least useful for the child because keeping only them leads to diverged training. The diverging results show that NN is not capable of providing all the needed capacity for the child, unlike the self-attention.

All in all, these experiments illustrate the robustness of the Transformer model in that it is able to train and reasonably well utilize parent weights even when the training is severely crippled. Interestingly, the attention is a crucial part of the network as it can compensate for the harmful effect to some extent. We use this knowledge in Section 5.4.3, where we evaluate the parent model with damaged word order.

Table 5.9: Results of child following a parent with swapped English side. "Baseline" is trained on child data only. "Aligned EN" is the more natural setup with English appearing on the "correct" side of the parent, the numbers in this column thus correspond to those in Table 4.10. The ‡ represents significantly better model when comparing "Transfer" and "Baseline".

Parent	Child	Transfer	Baseline	Aligned EN
Finnish→EN	EN→Estonian	**18.19** ‡	17.03	19.74
Russian→EN	EN→Estonian	**18.16** ‡	17.03	20.09
EN→Finnish	Estonian→EN	**22.75** ‡	21.74	24.18
EN→Russian	Estonian→EN	**23.12** ‡	21.74	23.54

5.4.2 Direction Swap in Parent and Child

In the previous section, we showed that the side of the network with shared language is modified the least, and when frozen, it leads to the best performance.

We experimented with scenarios of shared-source, shared-target, and no-shared language. In this section, we investigate the scenario, where the shared language is switched to the other side between parent and child in order to investigate if the technique can transfer other features across various parts of the network.

In other words, we now allow a mismatch in the translation direction of the parent and child. The parent XX→English is thus followed by an English→YY child or vice versa. We use Estonian–English language pair as the child with various parents. The results are from our paper Kocmi and Bojar (2018) ($T2T_4$).

This way, the child cannot use the parent target language model as languages on both sides changed. It is important to note that Transformer shares word embeddings for the source and target side. However, we showed in the previous section that the word embeddings are not crucial for the training, although some improvements could be due to better English embeddings.

The results in Table 5.9 document that an improvement can be reached even when none of the involved languages is reused on the same side. This suggests that the model can transfer further knowledge across various parts or layers of the network. Although there are too many factors that could influence it (language relatedness, parent training size, etc.), thus we cannot make any definite conclusion.

More importantly, the improvements are better when the shared language is aligned (column "Aligned EN"), which concludes that the shared language does play a significant role in transfer learning. This finding could be used whenever we want to train only one parent, for example, due to the time or resource restrictions and use it for more children even those with the shared language on the other side.

However, it cannot be the only source of improvements as we showed in Section 5.1.4 that the no-shared language scenario improves the performance of the child.

Table 5.10: Results of transfer learning with modified parent word order. The performance is measured in BLEU.

Parent	Child Performance	Parent Performance
Unmodified parent	20.07	23.48
Shuffle source	19.18	12.63
Shuffle target	19.16	2.78
Shuffle both	18.43	2.23
Sort target	19.45	2.29
Shuffled sentences	0.03	0.00
Baseline	15.80	–

Observation 33: *The improvements of transfer learning are partly but not fully attributed to the shared language between parent and child.*

5.4.3 Broken Word Order in Parent Model

We showed that the shared language plays a vital role in transfer learning, now we attempt to find some particular linguistic features explaining the gains.

Intuitively, the child model could transfer linguistic knowledge from the parent such as word order, length of the target language sentences, etc. In this section, we experiment with the somehow modified parent to study the effect on the transfer of knowledge.

We use the English→Czech as a parent model and the child is English→Estonian. We used the shared-source language scenario as it is harder for transfer learning (see Section 5.2), and the improvements cannot be attributed to the shared English target side as a better language model.

In the first experiment, we change the word order of a parent language pair in order to find out if the language word order plays an important role in transfer learning. Both of examined languages have mostly SVO (subject-verb-object) word order.

The word order is important for the attention mechanism that learns which parts of the source it should consider most.

We use sorting or shuffling of words (tokenized on whitespace) as a way to create a broken parent language. We modify only the word order of the parent model, therefore "shuffle source" means shuffling the source sentences (English in this experiment) and leaving the target language unmodified.

Additionally, we created an experiment, where we shuffled all sentences which breaks the sentence pairs (row "Shuffled sentences"). This way, the parent model could learn to generate random sentences that are not related to the source sentence. Thus it can mainly learn the decoder's language model.

Results ($T2T_{11}$) are tabulated in Table 5.10. The sorting of target side is less damaging than shuffling. Parent with both source and target side shuffled has the worst performance. However, it is still a good parent model for transfer

Table 5.11: Various automatic scores on English→Estonian testset. Scores prefixed "n" reported as $(1 - \text{score})$ to make higher numbers better.

Parent	BLEU	nPER	nTER	nCDER	chrF3	nCharacTER
Baseline	17.03	47.13	32.45	36.41	48.38	33.23
English→Russian	20.09	50.87	36.10	39.77	52.12	39.39
English→Czech	20.41	51.51	36.84	40.42	52.71	40.81

learning and improves the performance of the child be more than 2 BLEU points over the baseline.

Interestingly, the neural network has equal performance whenever the source and the target are shuffled despite the different performance of the parents (12.63 vs. 2.78 BLEU). We checked the performance of child model by various automatic metrics, and in all of them, the shuffled source and target reached similar scores.

This suggests that the word order of the target language is not the main feature in transfer learning. A better explanation is that the shuffling of either source or target breaks the attention mechanism in the same way, and then it needs to retrain on the child data. However, further analysis is needed as there can be other factors that can cancel each other out.

Observation 34: *There is little difference for the final child performance between shuffled word order of parent's source or target language.*

The poor performance of parent on English→Czech testset is understandable because the BLEU is computed based on n-grams of words that are heavily disrupted by shuffling. When we studied the outputs of the parent models, we noticed that they actually learned to translate and only the shuffle the output on top of that. The "sorted target" model translates and even alphabetically sorts the outputs. This documents the high flexibility of Transformer's skills in capturing word relations: between source and target, it happily resorts to lexical relations, and within the target, it easily captures (memorizes) alphabetical sorting.

We have noticed interesting behavior of "sort target". The model generates correct words in alphabetical order. We compute the unigram BLEU score and get 51.1 (compared to an unmodified model that has 53.8 unigram BLEU). If we evaluate the BLEU on the sorted target, we obtain 14.77 BLEU (result is not in the table). This result shows that the model learns to translate without access to the word order.

Observation 35: *NMT models, in general, can learn to some extent in the scenario with source sentences having broken word order. Although the word order is crucial for a good performance.*

We showed that damaging the word order in the parent model leads to a slight drop in performance of the child, still staying high above the baseline. In the next section, we analyze the outputs of the child model and look for

Table 5.12: Candidate total length, BLEU n-gram precisions and brevity penalty (BP). The reference length in the matching tokenization was 36062.

Parent	Length	BLEU Components	BP
Baseline	35326	48.1/21.3/11.3/6.4	0.979
English→Russian	35979	51.0/24.2/13.5/8.0	0.998
English→Czech	35921	51.7/24.6/13.7/8.1	0.996

potential over-estimations of translation quality that could emerge from the usage of BLEU metrics.

Lastly, "Shuffled sentences" cannot learn anything obtaining 0.03 BLEU. The parent model learned to generate one sentence for each input, and the child only changed the sentence into Estonian and generated "See on meie jaoks väga tähtis." (MT gloss: "This is very important to us.").

5.4.4 Output Analysis

We rely on automatic evaluation. Thus we need to prevent some potential over-estimations of translation quality due to BLEU. For this, we took a closer look at the baseline English→Estonian model (BLEU of 17.03 in Table 4.10 ($T2T_4$)) and two English→Estonian children derived from English→Czech (BLEU of 20.41) and English→Russian parent (BLEU 20.09).

Table 5.11 confirms that the improvements are not an artifact of uncased BLEU. The gains are apparent with several (now cased) automatic scores.

As documented in Table 5.12, the outputs of transferred models are slightly longer in terms of words produced. In the table, we also show individual n-gram precisions and Brevity Penalty (BP) of BLEU. The longer output helps to reduce the incurred BP, but the improvements are also apparent in n-gram precisions. In other words, the observed gain cannot be attributed solely to producing longer outputs.

Observation 36: *Transfer learning helps the child model to generate slightly longer sentences, and there are also clear improvements in produced n-grams.*

Table 5.13 explains the gains in unigram precisions by checking words in the child outputs that were present also in the baseline and/or confirmed by the reference. We see that about 44+20% of words of child outputs can be seen as "unchanged" compared to the baseline because they appear already in the baseline output. (The reference confirms the 44% words.)

The differing words are more interesting: "Neither" denotes the cases when the child model produced something different from the baseline and also from the reference. Gains in BLEU are due to "Reference only" words, i.e. words only in the child output and the reference but not in the baseline. For both parent setups, there are about 9–9.7 % of such words. We looked at these 3.2k and 3.5k words, and we have to conclude that these are regular Estonian words; no Czech or Russian leaks to the output and the gains are not due to simple word types common to all the languages (punctuation, numbers or named entities). We see

Table 5.13: Comparison of child outputs vs. the baseline and reference. Each column shows child trained from different parent either English→Russian or English→Czech.

Appeared in	English→Russian	English→Czech
Baseline+Reference	15902 (44.2 %)	15924 (44.3 %)
Baseline only	7209 (20.0 %)	7034 (19.6 %)
Reference only	3233 (9.0 %)	3478 (9.7 %)
Neither	9635 (26.8 %)	9485 (26.4 %)
Total	35979 (100.0 %)	35921 (100.0 %)

nearly identical BLEU gains even if we remove all such simple words from the child output and references.

5.4.5 Various Lengths of Parent Sentences

In Section 5.4.4, we showed that the child model generates slightly longer sentences than when trained without transfer learning. In this section, we investigate the effect is visible also when the parents are trained on corpora with sentences limited to certain length ranges.

In this experiment, we take the Czech–English corpus and randomly select sentences of various length creating training corpus for different parents. Each of the parents is trained with the corpus of sentences with lengths in the predefined range. The length of the source and target sentences are different. Therefore we use the sum of both lengths as the criterion.

We use four parent models. The first has sentence pairs with the length of either source or target of at most 10 words (tokenized by spaces). The second parent uses sentences of length 10 to 20 words. The third parent uses sentences of length 20 to 40, and the last parent uses sentences of length 40 to 60 words. We use English→Estonian as the child model in $T2T_{11}$ setup for this experiment.

We want to make the experiment comparable. Therefore each training corpus has exactly 300M words. Thus training set with the shortest sentences has the most sentences altogether.

Table 5.14 shows the results from our experiment. We start by discussing the parent performance. The parent model is unable to generalize from training data of different length for the training set. Although it generates sentences with slightly better length ratios than the training set, however, it cannot generalize the sentence length well. For example, when training on "1-10 words" it produces testset sentences with an average length of 10.9, which is more than it saw during the training. Similarly, "40-60 words" generates sentences of an average length of 35.5, which is shorter than training sentences.

It is an important finding that shows the need for checking training data in advance if they have similar length distribution as the target domain.

Observation 37: *NMT systems are highly influenced by the lengths of training sentences, and they are unable to generalize the sentence lengths.*

Table 5.14: Performance and average number of words generated over the testset. The references have average number of words 15.6 for the Czech testset 15.4 for the Estonian testset.

	Parent		Child	
Sentence lengths	BLEU	Avg. words	BLEU	Avg. words
1-10 words	8.57	10.9	16.57	15.3
10-20 words	16.21	15.4	17.48	15.3
20-40 words	12.59	21.9	17.99	15.3
40-60 words	5.76	35.5	16.80	15.5
1-60 words	22.30	15.3	19.15	15.4

When we discuss the performance of the child, we see that the child model generates sentences of valid length as the actual distribution over the testset is 15.4 words per test sentence. The performance is reaching 16.57 to 17.99 BLEU where the best parent model looks like the parent trained on "20-40 words" corpus. However, training the model on sentences of all lengths (row 1-60 words) leads to the best performance.

Observation 38: *Child model is not significantly influenced by the lengths of parent training sentences and can learn the length distribution by itself from child training data.*

Therefore the finding that child model generates slightly longer sentences than baseline from Section 5.4.4 is not due to the lengths of sentences in the parent training set.

In conclusion, it seems that the length of a parent model is not that important factor behind the performance improvements of the child model.

5.4.6 Parent's Performance Influence

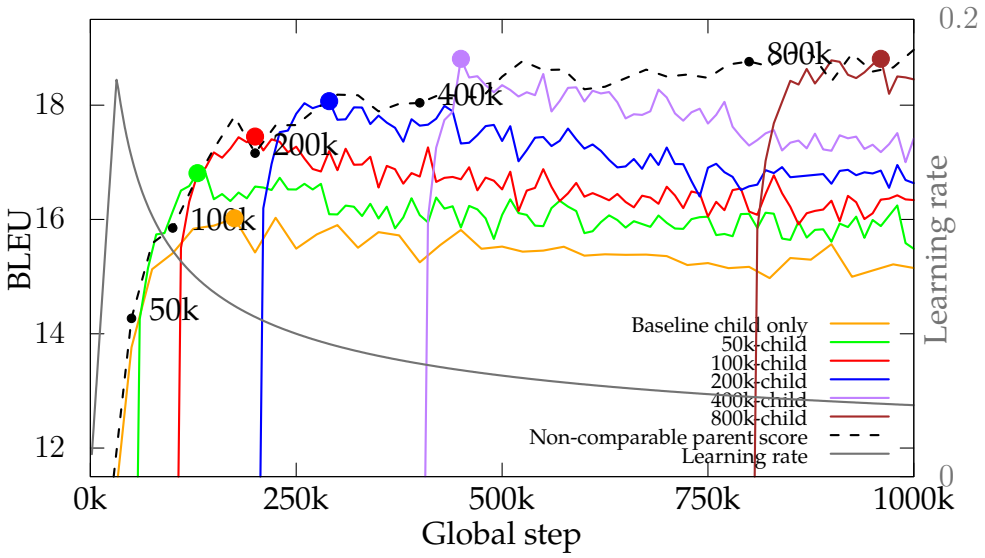
So far, we discussed features that could be considered linguistic. In the rest of the section, we examine various attributes that are associated with the neural architecture.

Transfer learning is, in fact, a way how to initialize weights for the child model in contrast to random initialization in the baseline. One of the explanations of the success could be attributed solely to the initialization or learning rate as these are only two attributes that change with the parent model.

In this section, we study the effect of various learning rates, and final parent performances on various models trained for a different amount of time in order to find out if there is a correlation between them and the child performance. Furthermore, we answer a question if it is necessary to train the parent model fully or is only a fragment of the training time enough?

In transfer learning, we train the parent model to its maximal performance, e.g. until convergence on the development set. In the following experiment, we compare the parent model in various stages of training and investigate the influence on the performance of the child model.

Figure 5.7: Learning curves on development set for English→Finnish parent and English→Estonian child. The child starts training after various number of parent's training steps. Black dots specify a performance of the parent at the moment when the child training started. The colored dots show the best performance of the child model. The grey curve shows, how the learning rate depends on the global step number.



In Kocmi and Bojar (2018) ($T2T_4$), we took English→Finnish as the parent model and used the warm-start transfer learning of child's English→Estonian model after 50k, 100k, 200k, 400k, and 800k of parent's steps. In order to simplify notation in this section, we label a child model that started after X parent's steps as a "X-child", e.g. 400k-child is a model trained after 400k parent's steps.

We conclude based on the learning curves in Figure 5.7 that the child's performance improvements correlate with the parent's performance. Furthermore, we see that only 50k steps are enough to outperform the baseline.

Observation 39: *Longer-trained parent model (or model trained until convergence) leads to a better performing child model.*

However, notice that the performance of the 400k-child is equal to the 800k-child, both having 18.8 BLEU despite the parent's performance improving from 18.0 to 18.8 BLEU.[4] This could be only an anomaly, or it could suggest that the child's performance also depends on different factors, which we try to examine further.

One such factor is the learning rate, which is the only parameter that depends on the global number step and decreases during the training. As the learning rate, we use a function of the inverse square root of steps with the warm-up stage of 16k steps, known as the "Noam" scheme. The steepness of the learning rate throughout the training is visualized in grey color in Figure 5.7.

[4]We remind that the BLEU scores of parent and child are not comparable as they are calculated on different testsets, and also that the scores fluctuate a lot between iterations.

Table 5.15: Experiment comparing various stages of parent model and learning rate. Each row corresponds to the same parent model as saved at given step (parent). Columns correspond to different learning rate shifts named by the global step at which it starts. The "0k-child" row is therefore baseline trained only on the child training set. The column "Parent" represent a performance of the parent at the step when the child was spawned.

Learning rate at:	0k	50k	100k	200k	400k	800k	1600k	Parent
0k-child	8.16	7.55	7.24	6.43	6.09	5.45	4.61	0.0
25k-child	12.02	12.47	12.76	12.95	12.97	12.91	12.98	17.7
50k-child	12.72	13.48	13.71	13.77	13.72	13.58	13.70	19.9
100k-child	13.31	14.01	14.06	14.26	14.64	14.59	14.56	21.6
200k-child	13.70	14.97	14.92	15.22	15.15	15.15	15.67	23.0
400k-child	14.16	15.20	15.89	15.89	16.08	15.83	15.84	23.9
800k-child	13.59	15.64	15.82	16.19	16.23	16.39	16.58	24.8
1600k-child	14.21	15.68	15.86	16.29	16.86	16.61	16.72	25.2

In order examine the factor of the learning rate, we prepare experiment where we fine-tune the child model on various stages of parent training as in Figure 5.7 however we investigate various learning rates for each parent step.

Initially, we wanted to fix the learning rate to a constant value. However, it would add a new factor to the question since even during the parent model training, the learning rate slowly decreases. Instead, we change the global step value before starting the child training to pretend that the parent was trained to a different stage than it actually was. The child learning rate follows the Noam learning curve. Thus we refer to the learning rate value using the global step index, e.g. 400k-child with 800k learning rate represent a model that fine-tunes on the parent model trained for 400k steps, and its learning rate behaves as if the parent was trained for 800k steps.

We now provide more details in the new experiment with slightly different settings. We decided to use the English→Czech as the parent model and English→Estonian as the child. We artificially downsampled the child training corpus Estonian–English to 100k sentences. We used Czech as a parent language pair in contrast to Finnish in Figure 5.7. We have used only 100k compared to 800k of child's training data in the original experiment. It was motivated to examine the effect of the less resourceful language. The initial learning rate is 0.2. The following experiment uses $T2T_{11}$.

Table 5.15 presents the results of the experiment; all results are evaluated on the same testset. Thus they are comparable, and we present them in the form of a heat-map for better visualization. However, the values within columns should be compared with care as they differ in the parent model that has been trained for a various number of steps. Note that the learning rate scheme of the parent model never changed.

From Table 5.15, we conclude that the learning rate schedule is important for the training. The 0k-child baseline trained without transfer learning (see the

first row in Table 5.15) has the best performance with learning rate starting at 0k. Therefore the initial warm-up steps (16k in total) are needed for proper training. In contrast, resetting learning rate to 0k across all the transfer setups (see the column "0k") harms the final child performance.

Observation 40: *Warm-up steps and the peak in learning rate are crucial for good performance of the baseline. However, repeating this peak in child training damages the performance of transfer learning.*

Beyond that, there is not a clear pattern of the best learning rate. For all transfer learning results it fluctuates between 200k and 1600k learning rate where the differences in BLEU are mostly not significant, we suppose that it is due to the shape of learning rate that is close to being constant. Therefore we conclude that the learning rate is not the primary source of improvements behind transfer learning.

Observation 41: *The improvements by transfer learning cannot be attributed to better chosen learning rate stage in its warmup-delay scheme.*

We see that with a better performing parent, the best performance of the child grows, contrarily to Figure 5.7 where the child performance has not changed between 400k-child and 800k-child. Therefore the parent performance plays an important role in transfer learning. The best child performance of 16.86 is obtained with the parent trained for 1600k steps, which is around three weeks of training on one GPU. In comparison, the average training of the child in this experiment was 50k steps. We use stopping criterion from Section 3.5.1 and saw all child models started to overfitting.

The improvements for the child model diminish in relation to the parent's step, for example, it takes only 400k steps to reach the performance of 16.08, but additional 1200k steps to improve by 0.72 BLEU. Interestingly, the 25k-child already outperforms the baseline. The 25k steps were trained for eight hours, thus proving that the improvements in transfer learning are usable after a short period of training the parent model.

To answer our initial questions, we conclude that child performance depends on the performance of the parent model. We showed that the learning rate is not the main factor of the transfer learning gains, but it can limit the maximum gains if changed. Moreover, we showed that transfer learning significantly improves child performance over the baseline, even when the parent has not been adequately trained.

5.4.7 Same Language Pair in Reverse Direction

We showed that transfer learning can extract knowledge from a parent model whenever the shared language is on the different translation side as well as improve the performance whenever both languages are different. In both cases, it is the additional data from the parent that probably affect the performance of the model. However, there is yet another explanation of the improvements: transfer learning improvements could be attributed to the better initialization of weights.

When we train the neural network, we have to decide on the initialization of the whole network. As Glorot and Bengio (2010); Mishkin and Matas (2016)

Table 5.16: Results of child following a parent with swapped direction. The ‡ represents significantly better results.

Parent	Child	Transfer	Child-only	Difference
EN→Estonian	Estonian→EN	**22.04** ‡	21.74	+0.30
Estonian→EN	EN→Estonian	**17.46**	17.03	+0.43
EN→Finnish	Finnish→EN	**20.23** ‡	19.90	+0.33
Finnish→EN	EN→Finnish	**14.51**	14.25	+0.16
EN→Odia	Odia→EN	**7.95** ‡	6.97	+0.98
Odia→EN	EN→Odia	**3.22**	3.19	+0.03
Spanish→French	French→Spanish	**28.54** ‡	27.89	+0.65
French→Spanish	Spanish→French	**27.69** ‡	27.21	+0.48

and we in Section 3.1.3 showed, NNs are sensitive to the variance of random initialization, and bad initialization can have a huge effect on the final model performance.

We can perceive transfer learning as a way of finding a better initialization of weights for the training of the child. Thus it could improve the performance mainly due to the effect of having weights initialized to better values.

In order to investigate this explanation, we prepare an experiment where the parent model does not have any additional training data. We train the parent on reversed training data than the child, in other words, the parent is XX→YY model, and the child is YY→XX. Thus the child model does not have access to any new training data. The experiments for Estonian–English are from our paper Kocmi and Bojar (2018) based on $T2T_4$, the rest are new in this thesis based on $T2T_{11}$. We selected languages to cover low-resource languages (Estonian–English) as well as high-resource languages (Spanish-French).

Table 5.16 shows a particularly exciting result: the parent does not use any other parallel sentences, but the very same corpus as a child with source and target side swapped and obtained a performance improvement. We see gains in both directions, although not always statistically significant.

Observation 42: *Transfer learning improves performance even in the scenario, where no new data are available, and the child is trained on parent training corpus in reverse direction.*

One explanation of the improvements could be that the training corpus is noisy and often contains English sentences on the wrong side, for example, in the Estonian part of the corpus. In order to verify it, we ran an automatic language identification and found out that Estonian part of the corpus contains only 0.1% English sentences, Finnish contains 3.4%, and Odia contains 0.0% of English sentences. The Estonian and Odia cannot be attributed to the noisiness as there is nearly zero English sentences and the 3.4% for Finnish is low that we do not think it is the main reason behind the improvements.

The low-resource language as Odia–English reached low performance due to insufficient data for model training (see Section 4.5.2). In contrast, the high-

resource language pair of French-Spanish reached significant improvements in both directions.

It is an exciting result. The model did not have access to any new data, yet it could extract new features from the reverse language pair, which it would not learn only from the original direction. Similar behavior has been shown in Niu et al. (2018), where they mixed both directions and added an artificial token indicating the target language.

The results from this experiment support our alternative explanation that the main improvements are simply from a better initialization of the model. Although, we showed that other features further improve the final performance of the model, such as shared language between parent and child, language relatedness, or size of parent training data.

5.5 Summarizing Transfer Learning Analysis

In this chapter, we analyzed various aspects of transfer learning. We tried to shed some light on the behavior of transfer learning as well as to peek inside the NMT black box. Although we made numerous observations and proposed several conclusions, we only scratched the surface, and much more work is needed in order to understand the behavior of the neural network.

At the beginning of the chapter in Section 5.1, we talked about the negative transfer and its effects on transfer learning in NMT. We found out that in NMT transfer learning the negative effects are limited in comparison to other fields where the negative transfer is a problem. We found out that using a parent model with less training data than the child can hurt the performance more than training from random initialization. On the other hand, transfer learning is especially useful for extremely low-resource child languages, and it does not produce relics from the parent target language.

In Section 5.2, we discussed the influence of the position of a shared language between parent and child. We found out that the shared-target language scenario converges faster and has a higher slope of the learning curve than the shared-source scenario. Both observations suggest that the network can learn more in the shared-target scenario. Furthermore, in the no-shared language scenario, the network does not forget the performance as quickly and can perform the parent translation to some extent, even with the final child model.

What is the relation between language relatedness and the parent training size was discussed in Section 5.3. We observed that relatedness does improve the performance of the child. However, the size of parent training data has a more substantial effect on the final performance of the child and even parent trained on a gibberish language can improve the performance of child more than a related language with less parallel sentences. Furthermore, we confirmed the finding by training the parent model on a vast corpus from mixed languages and obtained better results than high-resource parent Czech–English.

In the last section, we discussed if the gains can be mainly attributed to some linguistic features or if it is merely some better initialization of weights. We observed that the child is generating longer sentences, and we found out that word order and parent sentences length play a small role in transfer learning. We confirmed that the gains are not due to a better learning rate staging. However,

in the last experiment, we showed that the gains are seen even in the scenario, where the parent does not have any additional training data and is trained in reverse translation direction. We take it as the main hint that the gains are simply by a better initialization of the network.

In the next section, we provide a case study of using transfer learning in backtranslation and show that with our technique, the backtranslation can be used even on low-resource language pairs.

5.6 Case Study: Transfer Learning with Backtranslation

Let us conclude the analysis with a case study, where we apply transfer learning to the standard backtranslation approach for a low-resource language. The backtranslation approach relies on the performance of the initial model, and whenever the initial model has low performance, the backtranslation is not suitable, which is the case of low-resource languages. In this section, we show that transfer learning can help to overcome this problem.

This section is based on our paper Kocmi and Bojar (2019b) ($T2T_{11}$).

5.6.1 Backtranslation

The number of parallel sentences needed for training MT models is often very scarce. This is especially true for low-resource languages. However, the monolingual data are often more abundant. Many strategies have been used in MT in the past for employing resources from additional languages, see e.g. Wu and Wang (2007), Nakov and Ng (2012), El Kholy et al. (2013), or Hoang and Bojar (2016).

The approaches of using monolingual data to improve MT date back to SMT where they were used to improve the language model (Brants et al., 2007) or the translation model (Schwenk, 2008; Bertoldi and Federico, 2009; Bojar and Tamchyna, 2011). A similar technique of improving only the target language model was examined in early NMT (Gulcehre et al., 2015).

Sennrich et al. (2016a) took a different approach. They showed that creating synthetic parallel sentences by translation of monolingual text and using this synthetic corpus as a training set leads to significant improvements in performance. This approach quickly gained popularity, and it is considered the current best practice of using monolingual data in NMT.

The technique of backtranslation, as the method is named, consists of first training a system in the reverse direction on human-generated "authentic" parallel sentences, i.e. target language to source language. This system is then used to translate monolingual data of a target language. The resulting translations and their monolingual counterparts are used as additional training data for the training of the final model in the original translation direction. Thus we can consider the backtranslation as a way of transferring knowledge learned in one direction to the reverse direction through generated data.

Sennrich et al. (2016a) propose two regimes of incorporating synthetic training data. Either as a fine-tuning a general model or training the model on a

mixed corpus. The former method uses a general model trained on authentic data that is fine-tuned by 1:1 mix of authentic and synthetic data. The latter method by training the model from beginning on the mix of the authentic and synthetic data. The second approach became the standard approach in the NMT systems, as can be seen annually in the WMT Translation Shared Tasks by competing systems (Bojar et al., 2019).

Popel (2018a) investigated yet another variation of incorporating synthetic and authentic data. He proposed to switch iteratively between synthetic and authentic data without mixing them. This approach, called concat-regime backtranslation outperformed the training on the mixture.

Furthermore, backtranslation can be used as a domain adaptation when a limited number of in-domain parallel sentences exists (Chinea-Rios et al., 2017; Chu et al., 2018), and the in-domain monolingual data are available.

Lastly, Hoang et al. (2018) showed a way of improving the backtranslation by repeating the backtranslation process. They presented a method of training two NMT systems in parallel, each in the opposite translation direction of a language pair, which alternately generate synthetic data for the reverse system to improve. Hoang et al. (2018) concluded that the second round of backtranslation improves performance. However, the third round seems not to lead to any significant improvements.

5.6.2 Backtranslation with Transfer Learning

Backtranslation is helpful mainly in scenarios where the parent model, which is used to translate monolingual data, has a reasonable performance (Hoang et al., 2018; Bawden et al., 2019). However, for low-resource language as it is hard to train any suitable initial model.

We propose to use transfer learning on the low-resource language pair for training the initial model with a reasonable score. We train two models in parallel, one for each translation direction. The models iteratively generate backtranslated data for the other one. We show this approach on two low-resource languages Gujarati–English and Kazakh–English.

As a parent model we use Czech–English for Gujarati–English and Russian–English for Kazakh–English. The training procedure is as follows.

First, we train two high-resource parent models for each studied language until convergence: English→Czech, Czech→English, English→Russian and Russian→English.

Then we apply transfer learning with the use of an authentic dataset of the corresponding low-resource language pair. We preserve the English side: Czech→English serves as the parent to Gujarati→English and English→Czech to English→Gujarati. The same strategy is used for transfer learning from Russian to Kazakh.

After transfer learning, we select one of the translation directions to translate monolingual data (model ①). As the starting system for the backtranslation process, we have selected the English→Gujarati and Kazakh→English. The decision for Kazakh–English is motivated by choosing the better performing model, see Table 5.17. This is however only a rough estimate because higher BLEU scores across various language pairs do not always need to indicate better perfor-

Table 5.17: Testset BLEU scores of our setup. Except for the baseline, each column shows improvements obtained after fine-tuning a single model on different datasets beginning with the score on a trained parent model. The circled names points to the systems in the right side of the table.

Training dataset	EN→GU	GU→EN	EN→KK	KK→EN
Authentic (baseline)	2.0	1.8	0.5	4.2
Parent dataset	0.7	0.1	0.7	0.6
Authentic (transfer learning)	① 9.1	9.2	6.2	① 14.4
Synth generated by model ①	–	② 14.2	② 8.3	–
Synth generated by model ②	③ 13.4	–	–	17.3
Synth generated by model ③	–	④ 16.2	–	–
Synth generated by model ④	13.7	–	–	–
Averaging + beam 8	14.3	17.4	8.7	18.5

mance; the properties of the target language such as its morphological richness affect the absolute value of the score. For the Gujarati–English, we decided to start with the model with English source side in contrast to Kazakh→English.

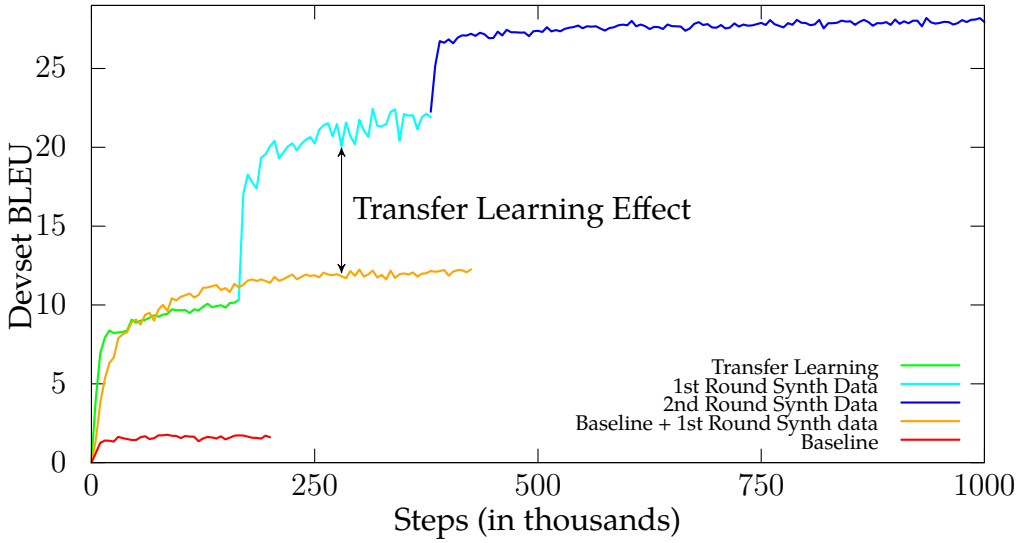
After the backtranslation, we mix the synthetic data with the authentic parallel corpus and train the first backtranslated model ②. We repeat this process: Use the improved system (③ and then ④) to backtranslate the monolingual data, and use this data in order to train the improved system in the reverse direction. We make two rounds of backtranslation for both directions on Gujarati–English and only one round of backtranslation on Kazakh–English.

The baseline models in Table 5.17 are trained on the authentic data only, and it seems that the number of parallel sentences is not sufficient to train the NMT model for the investigated language pairs (we obtained performance of 0.5 to 4.2 BLEU). The remaining rows show incremental improvements as we perform the various training stages. The last stage of model averaging takes the best performing model and averages it with the previous seven checkpoints that are one and a half hours of training time from each other.

Figure 5.8 above shows the progress of training of Gujarati–English models in both directions. We can notice that after each change of parallel corpus, there is a substantial improvement in the performance. The learning curve is computed on the development data. The corresponding scores for the testsets are in Table 5.17.

We also run a standard approach of backtranslation without transfer learning, where we first trained baseline model to translate monolingual data and then trained a model (in reverse direction) on those synthetic data with concatenation with authentic data. We visualize the training with orange in Figure 5.8. When comparing with a transferred model trained with the first round of back-

Figure 5.8: Learning curves of Gujarati→English models. Our approach is combination of four steps. The first is to train the parent model for 2000k steps (not in the figure), then transfer learning (green curve). Then we continue with two rounds of backtranslated data (both blue curves). Baseline without transfer learning and backtraslation is in red. Standard approach of backtranslation without transfer learning is in orange.



translation data (light blue), we can see the clear improvement in performance due to transfer learning.[5]

Observation 43: *Transfer learning can be used as an initial step for the low-resource NMT training with backtranslation technique to largely improve the performance.*

In conclusion, we showed that transfer learning can be used in combination with other techniques. Furthermore, it can be used to generate a reasonably good initial model for backtranslation technique in the low-resource scenario, where it is hard to train the model from random initialization.

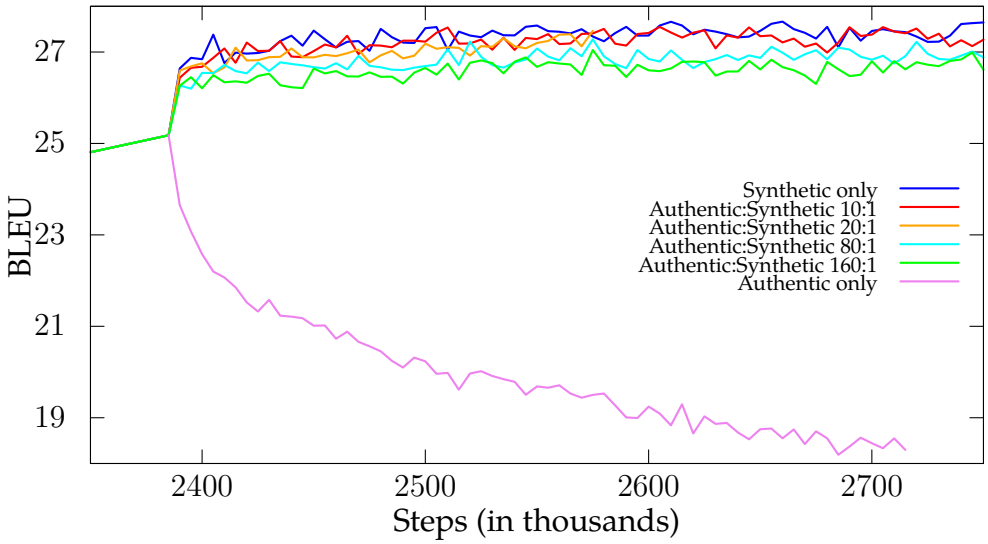
5.6.3 Ratio between Authentic and Synthetic

Backtranslation as a way of transferring knowledge through data to the other direction can be used in theory on as much monolingual data as accessible. For example, English has billions of monolingual data. Thus we could create a training corpus for translation from any language to the English with billions of sentences for all languages. However, this corpus would not contain the authentic data in 1:1 ratio as proposed by Sennrich et al. (2016a).

Edunov et al. (2018) showed that the shortage of authentic data in the synthetic corpus could be supplemented by oversampling the authentic data.

[5]Each scenario (orange/light blue) use synthetic data generated by different model from an equal monolingual corpus. The model generating the synthetic data either used transfer learning (light blue) or did not (orange).

Figure 5.9: Comparison of various ratio between the synthetic and authentic sentences.



Poncelas et al. (2018) investigated the effect of various ratios between authentic and synthetic training data on German→English scenario. The authors suggest the ratio of 1:2 in favor of synthetic data. The higher ratio of synthetic data does not help but also does not decrease performance significantly. The authors have not experimented with oversampling of authentic data.

To conclude their findings, we can use a synthetic corpus of any size as long as we mix the authentic data into the corpus. In case that we lack a large quantity of authentic data, we can oversample the available ones. Nonetheless, they experimented with the high-resource language only. Hence we evaluated their findings on a low-resource scenario.

We experimented with Gujarati–English language pair, which has only 172k parallel sentences. We used a warm-start transfer learning and two rounds of backtranslation as proposed by Hoang et al. (2018). During the second round of training on the backtranslated data, we evaluated various ratios between the synthetic and authentic data.

In order to report the ratio between authentic and synthetic, we backtranslated 3.6M English sentences into Gujarati, which is exactly twenty times more monolingual data than the size of authentic data. Then we oversampled the authentic data and mixed them to the synthetic corpus to get final training set. For example, the ratio "authentic:synthetic 10:1" means that the authentic has been multiplied ten times, which to match half size of the synthetic sentences.

In Figure 5.9, we can see the difference in performance when comparing the ratio between the amount of synthetic and authentic data. The model learning curves start at the 2.4M step as a visualization of their previous transfer learning and the first round of backtranslation (see Figure 5.8).

Using only low-resource authentic parallel sentences drastically damages the performance of an already good model. Thus the previous training on synthetic data is necessary. However, the most surprising fact is that adding

more authentic data damages the performance, which is in contrast to claims of Poncelas et al. (2018) and Edunov et al. (2018).

Observation 44: *Oversampling authentic data to match synthetic data hurts NMT performance.*

To explain this phenomenon, we need to take into account that in order to match the ratio, we oversampled the authentic data multiple times. For example, the worst-performing ratio of "160:1" had 160 copies of authentic data within, which could lead to overfitting on them. Thus we believe that the authentic data are useful in synthetic corpus only to some extent and after having much more synthetic data, it is better to use only the synthetic high number of parallel sentences instead of adding more and more copies of authentic data in order to match a ratio between authentic and synthetic data. We would like to understand where the tipping point is and if we can add more monolingual data and obtain a better performance. However, it is over the scope of this thesis, and we leave it as an open question.

This experiment once again supported our intuition that the more data NMT has, the better performance it reaches even though they are artificially created, even to the extent that the authentic data become useless.

In conclusion, we showed that transfer learning can be combined with other techniques. It substantially helps when used with backtranslation. Furthermore, we showed that oversampling authentic data (Edunov et al., 2018) is not useful for low-resource languages and can lead to obtaining lower performance than without oversampling.

6 Conclusion

As stated in the introduction, this thesis had two main parts. The first was the introduction of various techniques for transfer learning in NMT. The second was a broad analysis of transfer learning behavior.

We presented methods for both the cold-start and warm-start scenario, where both of them outperform a baseline that is trained without transfer learning. We showed that transfer learning helps for low-resource as well as high-resource languages.

In the cold-start scenario, we presented Direct Transfer, a technique that trains a child model without any modifications, and Transformed Vocabulary, a technique that adapts a parent vocabulary for the need of the child by randomly overriding unused embeddings. Furthermore, we showed a proof-of-concept of training a model from the parent that has not been trained by us. This concept opens doors to better replicability of experiments. With this technique, researchers could "never train their models from scratch ever again".[1]

In the warm-start scenario, we proposed two approaches: Merged and Balanced Vocabulary. Both of them prepared vocabulary in advance of the parent training. We showed significant gains even over the cold-start technique.

The second part of this thesis is the analysis of transfer learning behavior. We shed some light on transfer learning in NMT. We studied the phenomenon of negative transfer. We showed that transfer learning behaves differently concerning the position of the shared language. Moreover, we studied various aspects of transfer learning, and we concluded that the main gains are due to the size of the parent model. Furthermore, we believe that transfer learning behaves as a better initialization method to some extent.

All our observations can be found in the *List of Observations* on page 141. However, our study is only a small step in understanding transfer learning or even neural networks.

[1] This statement is obviously a bit exaggerated.

Table 6.1: The total amount of time spent, energy consumed, and CO_2 emitted during our transfer learning research. The numbers are only a rough estimate.

GPU type	Time spent	Energy Consumption	CO_2 Emissions
Tesla K40c	3658 hours	1.0 GWh	0.5 t
GeForce GTX 1080	2221 hours	0.6 GWh	0.3 t
GeForce GTX 1080 Ti	52227 hours	14.1 GWh	7.2 t
Quadro P5000	14996 hours	3.2 GWh	1.7 t
Total	73102 hours	18.9 GWh	9.8 t

6.1 Ecological Trace

Earlier this year, Strubell et al. (2019) published work on the impact of deep learning on CO_2 emissions. They estimated that a single Transformer architecture hyper-parameter search produced 284 tonnes of CO_2.[2] Many researchers pointed out that it reports numbers based on the U.S. energy mix. However, Google claims that its platform is 100% renewable.[3] Moreover, the most carbon-intensive scenario in the study costs between $1 million and $3 million (Strubell et al., 2019), which is not an everyday expense.

We believe that it is important to raise awareness of and quantify the potential CO_2 impact of deep learning. Thus we try to calculate the impact of this study. We roughly calculate the wall-clock time, power consumption, and CO_2 emissions.

We start by calculating the total wall-clock time on GPUs. Our cluster saves information about GPU usage every 10 minutes. We recovered logs from the last 20 months (the time when almost all of our transfer learning experiments have been done) and calculated how many GPU hours we spent on our experiments.

The average power consumption is harder to calculate. Based on discussion with our IT department, the average GeForce takes 200W per hour and Quadro P5000 160W per hour, when entirely in use. We need to take into account also the air-conditioning of the room, which can be roughly calculated by coefficient 1.3 to 1.4, we take the 1.35 as a middle. With this in mind, we use 270W for Tesla and GeForce cards and 216W for Quadro P5000.

The last step is the calculation of CO_2 emissions. Based on the report by OECD (2015), the average CO_2 emissions during the production of electricity in the Czech Republic is 516g per kWh (in the year 2015).

Table 6.1 represent the total number of hours, energy consumption and CO_2 emissions for our experiments. An important notice is that energy consumption is *very roughly* estimated. It is based on GPU in full use, and the effect of air-conditioning is only estimated. Therefore the numbers are more likely the upper bound.

[2]The paper reports numbers in imperial units, for which it was criticized. We recomputed the numbers for SI units.

[3]Source: `https://cloud.google.com/sustainability/`

Table 6.2: Comparison of CO_2 emissions from various sources. The numbers are from (Strubell et al., 2019).

GPU type	CO_2 Emissions
Our research	9.8 t
Air travel NY $\leftrightarrow$ SF per passenger	0.9 t
Average person production in year	5.0 t
Average American person production in year	16.4 t
Lifetime car consumption	57.2 t

Our experiments produced a similar amount of CO_2 as eleven round-trip-flights from New York to San Francisco for one passenger or as an average person produces within two years. For more comparisons, see Table 6.2.

Nonetheless, we proposed a way that reduces the total training time. Primarily the cold-start scenario can be used as a technique, which can be used to improve performance and lower the total training time.

Furthermore, machine learning can help to tackle climate change in various ways: predicting the electricity demand; flexibly managing household, commercial or electric vehicle demand; optimizing transportation routes; forecasting extreme climate events; modeling disease spreading; and many more as summarized by Rolnick et al. (2019).

In conclusion, deep learning is energy-intensive, and further research is needed to find the best ways to minimize the impact. However, restrictions on research are not the answer to the problem. The main issue is the generation of electricity, which for most of CO_2 emission in the Czech Republic is in coal power plants. As mentioned earlier, Google claims that its AI cluster is 100% run by renewable energy; Amazon claims 50%.[4]

6.2 Final Words

I am fortunate to have gone through an exciting journey of studying NMT since its early days of becoming the standard approach in MT. And I hope this dissertation thesis will provide useful background and inspiration for future research in NMT transfer learning.

Lastly, it would mean a lot to me if you would give me feedback or a short comment at `http://kocmi.tk/thesis/`.

[4]Source: `https://aws.amazon.com/about-aws/sustainability/`

Bibliography

Shawki A Al-Dubaee, Nesar Ahmad, Jan Martinovic, and Vaclav Snasel. Language identification using wavelet transform and artificial neural network. In *Computational Aspects of Social Networks (CASoN), 2010 International Conference on*, pages 515–520. IEEE, 2010.

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

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

Mohammad Taha Bahadori, Yan Liu, and Dan Zhang. A general framework for scalable transductive transfer learning. *Knowledge and information systems*, 38 (1):61–83, 2014.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR 2015*, 2014.

Mona Baker et al. Corpus linguistics and translation studies: Implications and applications. *Text and technology: In honour of John Sinclair*, 233:250, 1993.

Timothy Baldwin and Marco Lui. Language identification: The long and the short of the matter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 229–237, Los Angeles, California, June 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N10-1027.

Rachel Bawden, Nikolay Bogoychev, Ulrich Germann, Roman Grundkiewicz, Faheem Kirefu, Antonio Valerio Miceli Barone, and Alexandra Birch. The university of edinburgh submissions to the wmt19 news translation task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 103–115, Florence, Italy, August 2019. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W19-5304.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3 (Feb):1137–1155, 2003.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

Alexandre Bérard, Christophe Servan, Olivier Pietquin, and Laurent Besacier. Multivec: a multilingual and multilevel representation learning toolkit for nlp. In *The 10th edition of the Language Resources and Evaluation Conference (LREC 2016)*, May 2016.

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, Athens, Greece, March 2009. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W09-0432.

Ondřej Bojar and Aleš Tamchyna. Improving translation model by monolingual data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 330–336, Edinburgh, Scotland, July 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W11-2138.

Ondřej Bojar, Kamil Kos, and David Mareček. Tackling sparse data issue in machine translation evaluation. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 86–91, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P10-2016.

Ondřej Bojar, Zdeněk Žabokrtský, Ondřej Dušek, Petra Galuščáková, Martin Majliš, David Mareček, Jiří Maršík, Michal Novák, Martin Popel, and Aleš Tamchyna. The Joy of Parallelism with CzEng 1.0. In *Proceedings of LREC2012*, Istanbul, Turkey, May 2012. ELRA, European Language Resources Association. In print.

Ondřej Bojar. *The Oxford handbook of inflection*, chapter Machine Translation. Oxford University Press, USA, 2015.

Ondřej Bojar, Matouš Macháček, Aleš Tamchyna, and Daniel Zeman. Scratching the surface of possible translations. In *Text, Speech and Dialogue: 16th International Conference, TSD 2013. Proceedings*, pages 465–474, Berlin / Heidelberg, 2013. Springer Verlag. ISBN 978-3-642-40584-6.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal, September 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/W15-3001.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2301.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. Findings of the 2017 conference on machine translation (wmt17). In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 169–214, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W17-4717.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 272–307, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6401.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Christof Monz, Mathias Müller, and Matt Post. Findings of the 2019 conference on machine translation (wmt19). In *Proceedings of the Fourth Conference on Machine Translation, Volume 2: Shared Task Papers*, Florence, Italy, August 2019. Association for Computational Linguistics.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large Language Models in Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D07-1090.

Lorenzo Bruzzone and Mattia Marconcini. Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):770–787, 2009.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluation the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006. URL https://www.aclweb.org/anthology/E06-1032.

Simon Carter, Wouter Weerkamp, and Manos Tsagkias. Microblog language identification: overcoming the limitations of short, unedited and idiomatic text. *Language Resources and Evaluation*, 47(1):195–215, 2013. ISSN 1574-0218. doi: 10.1007/s10579-012-9195-y. URL http://dx.doi.org/10.1007/s10579-012-9195-y.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

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

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1179.

Christos Christodouloupoulos and Mark Steedman. A massively parallel corpus: the bible in 100 languages. *Language resources and evaluation*, 49(2):375–395, 2015.

Chenhui Chu and Rui Wang. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1111.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. An empirical comparison of domain adaptation methods for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–391, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-2061. URL https://www.aclweb.org/anthology/P17-2061.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. A comprehensive empirical comparison of domain adaptation methods for neural machine translation. *Journal of Information Processing*, 26:529–538, 2018.

Raj Dabre, Tetsuji Nakagawa, and Hideto Kazawa. An empirical study of language relatedness for transfer learning in neural machine translation. In *Proceedings of the 31st Pacific Asia Conference on Language, Information and Computation*, pages 282–286, 2017.

Michael Denkowski and Graham Neubig. Stronger baselines for trustable results in neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 18–27, Vancouver, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-3203. URL https://www.aclweb.org/anthology/W17-3203.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N19-1423.

Bhuwan Dhingra, Hanxiao Liu, Ruslan Salakhutdinov, and William W. Cohen. A comparative study of word embeddings for reading comprehension. *CoRR*, abs/1703.00993, 2017. URL http://arxiv.org/abs/1703.00993.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1166. URL https://www.aclweb.org/anthology/P15-1166.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding Back-Translation at Scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-1045.

Ahmed El Kholy, Nizar Habash, Gregor Leusch, Evgeny Matusov, and Hassan Sawaf. Language independent connectivity strength features for phrase pivot statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P13-2073.

Denis Emelin, Ivan Titov, and Rico Sennrich. Widening the representation bottleneck in neural machine translation with lexical shortcuts. In *Proceedings of the Fourth Conference on Machine Translation*, pages 102–115, Florence, Italy, August 2019. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W19-5211.

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.

Nazli Farajidavar, Teofilo de Campos, and Josef Kittler. Transductive transfer machine. In Daniel Cremers, Ian Reid, Hideo Saito, and Ming-Hsuan Yang, editors, *Computer Vision – ACCV 2014*, pages 623–639, Cham, 2015. Springer International Publishing. ISBN 978-3-319-16811-1.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California, June 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N16-1101.

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.

Markus Freitag and Yaser Al-Onaizan. Fast domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06897*, 2016.

Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2): 23–38, 1994.

Petra Galuščáková and Ondřej Bojar. Improving smt by using parallel data of a closely related language. In *Proc. of HLT*, pages 58–65, 2012.

Jianfeng Gao and Min Zhang. Improving language model size reduction using better pruning criteria. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 176–182. Association for Computational Linguistics, 2002.

Archana Garg, Vishal Gupta, and Manish Jindal. A survey of language identification techniques and applications. *Journal of Emerging Technologies in Web Intelligence*, 6(4):388–400, 2014.

Liang Ge, Jing Gao, Hung Ngo, Kang Li, and Aidong Zhang. On handling negative transfer and imbalanced distributions in multiple source transfer learning. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 7 (4):254–271, 2014.

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*, ICML'17, pages 1243–1252. JMLR.org, 2017. URL http://dl.acm.org/citation.cfm?id=3305381.3305510.

Martin Gellerstam. Translationese in swedish novels translated from english. *Translation studies in Scandinavia*, 1:88–95, 1986.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520, 2011.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. Can machine translation systems be evaluated by the crowd alone. *Natural Language Engineering*, 23(1):3–30, 2017.

Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O.K. Li. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 344–354, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1032. URL https://www.aclweb.org/anthology/N18-1032.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*, 2015.

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. URL http://arxiv.org/abs/1803.05567.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of EAMT*, volume 2005, pages 133–142, 2005.

Geoffrey Hinton, Jeff Dean, and Oriol Vinyals. Distilling the knowledge in a neural network. *NIPS 2014*, 2014.

Duc Tam Hoang and Ondřej Bojar. Pivoting methods and data for czech-vietnamese translation via english. *Baltic Journal of Modern Computing*, 4 (2):190–202, 2016.

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

Eduard Hovy, Maghi King, and Andrei Popescu-Belis. An introduction to mt evaluation. In *Proceedings of Machine Translation Evaluation: Human Evaluators meet Automated Metrics. Workshop at the LREC 2002 Conference. Las Palmas, Spain*, pages 1–7, 2002.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1001. URL https://www.aclweb.org/anthology/P15-1001.

Serena Jeblee, Weston Feely, Houda Bouamor, Alon Lavie, Nizar Habash, and Kemal Oflazer. Domain and dialect adaptation for machine translation into egyptian arabic. *ANLP 2014*, page 196, 2014.

Melvin Johnson, Mike Schuster, Quoc Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernand a 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 Linguistics*, 5:339–351, 2017. ISSN 2307-387X. URL https://transacl.org/ojs/index.php/tacl/article/view/1081.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://www.aclweb.org/anthology/D14-1181.

Yunsu Kim, Yingbo Gao, and Hermann Ney. Effective cross-lingual transfer of neural machine translation models without shared vocabularies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1246–1257, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1120.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, volume 4, pages 388–395, 2004.

Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, August 2017. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W17-3204.

Philipp Koehn, Franz J. Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133, 2003. URL https://www.aclweb.org/anthology/N03-1017.

Philipp Koehn, Huda Khayrallah, Kenneth Heafield, and Mikel L. Forcada. Findings of the wmt 2018 shared task on parallel corpus filtering. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 739–752, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6454.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-2012.

Siddharth Krishna Kumar. On weight initialization in deep neural networks. *arXiv preprint arXiv:1704.08863*, 2017.

Surafel M Lakew, Aliia Erofeeva, Matteo Negri, Marcello Federico, and Marco Turchi. Transfer learning in multilingual neural machine translation with dynamic vocabulary. *IWSLT. Bruges, Belgium*, 2018.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1030. URL https://www.aclweb.org/anthology/N16-1030.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-1549.

Alon Lavie and Abhaya Agarwal. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W07-0734.

Will Lewis, Robert Munro, and Stephan Vogel. Crisis mt: Developing a cookbook for mt in crisis situations. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, July 2011. URL https://www.microsoft.com/en-us/research/publication/crisis-mt-developing-a-cookbook-for-mt-in-crisis-situations/.

Xiaoqing Li, Jiajun Zhang, and Chengqing Zong. One sentence one model for neural machine translation. In *Proceedings of the 11th Language Resources and Evaluation Conference*, Miyazaki, Japan, May 2018. European Language Resource Association. URL https://www.aclweb.org/anthology/L18-1146.

Jindrich Libovický and Jindrich Helcl. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3016–3021. Association for Computational Linguistics, 2018. URL https://www.aclweb.org/anthology/D18-1336/.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586*, 2015.

Yichao Lu, Phillip Keung, Faisal Ladhak, Vikas Bhardwaj, Shaonan Zhang, and Jason Sun. A neural interlingua for multilingual machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 84–92, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6309.

Marco Lui and Timothy Baldwin. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P12-3005.

Marco Lui, Jey Han Lau, and Timothy Baldwin. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2:27–40, 2014.

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.

Minh-Thang Luong and Christopher D. Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1100. URL https://www.aclweb.org/anthology/P16-1100.

Qingsong Ma, Ondřej Bojar, and Yvette Graham. Results of the wmt18 metrics shared task: Both characters and embeddings achieve good performance. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 682–701, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6451.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. URL https://www.aclweb.org/anthology/J93-2004.

Antonio Valerio Miceli Barone, Barry Haddow, Ulrich Germann, and Rico Sennrich. Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1494, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1156.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL http://arxiv.org/abs/1301.3781.

Dmytro Mishkin and Jiri Matas. All you need is a good init. *4th International Conference on Learning Representations*, 2016.

Saif M Mohammad, Mohammad Salameh, and Svetlana Kiritchenko. How translation alters sentiment. *J. Artif. Intell. Res.(JAIR)*, 55:95–130, 2016.

Cecilia Montes-Alcalá. Blogging in Two Languages: Code-Switching in Bilingual Blogs. In Jonathan Holmquist, Augusto Lorenzino, and Lotfi Sayahi, editors, *Selected Proceedings of the Third Workshop on Spanish Sociolinguistics*, pages 162–170. Cascadilla Proceedings Project, Somerville, MA, USA, 2007. ISBN 978-1-57473-418-8.

Seungwhan Moon and Jaime G Carbonell. Completely heterogeneous transfer learning with attention-what and what not to transfer. In *IJCAI*, pages 2508–2514, 2017.

Preslav Nakov and Hwee Tou Ng. Improved statistical machine translation for resource-poor languages using related resource-rich languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1358–1367, Singapore, August 2009. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D09-1141.

Preslav Nakov and Hwee Tou Ng. Improving statistical machine translation for a resource-poor language using related resource-rich languages. *Journal of Artificial Intelligence Research*, 44:179–222, 2012.

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, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D18-1103.

Toan Q. Nguyen and David Chiang. Transfer learning across low-resource, related languages for neural machine translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 296–301. Asian Federation of Natural Language Processing, 2017. URL http://aclweb.org/anthology/I17-2050.

Jan Niehues, Roldano Cattoni, Stüker Sebastian, Mauro Cettolo, Marco Turchi, and Marcello Federico. The iwslt 2018 evaluation campaign. In *International Workshop on Spoken Language Translation*, pages 2–6, 2018.

Xing Niu, Michael Denkowski, and Marine Carpuat. Bi-directional neural machine translation with synthetic parallel data. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 84–91, Melbourne, Australia, July 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-2710.

OECD. *Climate Change Mitigation*. The Organisation for Economic Co-operation and Development, 2015. doi: https://doi.org/https://doi.org/10.1787/9789264238787-en. URL https://www.oecd-ilibrary.org/content/publication/9789264238787-en.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010. ISSN 1041-4347. doi: 10.1109/TKDE.2009.191.

Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

Shantipriya Parida, Ondřej Bojar, and Satya Ranjan Dash. Odiencorp: Odia-english and odia-only corpus for machine translation. In *Smart Computing and Informatics*. Springer, 2018.

Pavel Pecina. Adaptation of machine translation to specific domains and applications. *Univerzita Karlova*, 2017.

Alberto Poncelas, Dimitar Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. Investigating backtranslation in neural machine translation. *arXiv preprint arXiv:1804.06189*, 2018.

Martin Popel. Machine translation using syntactic analysis. *Univerzita Karlova*, 2018a.

Martin Popel. Cuni transformer neural mt system for wmt18. In *Proceedings of the Third Conference on Machine Translation*, pages 486–491, Belgium, Brussels, October 2018b. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-64051.

Martin Popel and Ondřej Bojar. Training Tips for the Transformer Model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70, 2018. URL https://content.sciendo.com/view/journals/pralin/110/1/article-p43.xml.

Matt Post. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels, oct 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6319.

Lorien Y Pratt, Jack Mostow, Candace A Kamm, and Ace A Kamm. Direct transfer of learned information among neural networks. In *AAAI*, volume 91, pages 584–589, 1991.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana, June

2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2084. URL https://www.aclweb.org/anthology/N18-2084.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

David Rolnick, Priya L Donti, Lynn H Kaack, Kelly Kochanski, Alexandre Lacoste, Kris Sankaran, Andrew Slavin Ross, Nikola Milojevic-Dupont, Natasha Jaques, Anna Waldman-Brown, et al. Tackling climate change with machine learning. *arXiv preprint arXiv:1906.05433*, 2019.

Cynthia Rudin. Please stop explaining black box models for high stakes decisions. *NIPS 2018 Workshop on Critiquing and Correcting Trends in Machine Learning*, 2018.

Inaki San Vicente, Iker Manterola, et al. Paco2: A fully automated tool for gathering parallel corpora from the web. In *LREC*, pages 1–6, 2012.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

Holger Schwenk. Investigations on large-scale lightly-supervised training for statistical machine translation. In *International Workshop on Spoken Language Translation (IWSLT) 2008*, 2008.

Rico Sennrich and Biao Zhang. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany, August 2016a. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P16-1009.

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 (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016b. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P16-1162.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In *HLT-NAACL*, pages 35–40, 2016c.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 371–376, Berlin, Germany, August 2016d. Association for Computational Linguistics. doi: 10.18653/v1/W16-2323. URL https://www.aclweb.org/anthology/W16-2323.

Christophe Servan, Josep Crego, and Jean Senellart. Domain specialization: a post-training domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06141*, 2016.

Christopher J Shallue, Jaehoon Lee, Joe Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E Dahl. Measuring the effects of data parallelism on neural network training. *arXiv preprint arXiv:1811.03600*, 2018.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. *arXiv preprint arXiv:1804.04235*, 2018.

F Simons. *Ethnologue: 21st Edition*. SIL International, Dallas, Texas, 2018.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2627435.2670313.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1355.

Sara Stymne. The effect of translationese on tuning for statistical machine translation. In *The 21st Nordic Conference on Computational Linguistics*, pages 241–246, 2017.

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.

Aleš Tamchyna, Marion Weller-Di Marco, and Alexander Fraser. Modeling target-side inflection in neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 32–42, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4704. URL https://www.aclweb.org/anthology/W17-4704.

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, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6313.

Jörg Tiedemann. Character-based psmt for closely related languages. In *Proceedings of the 13th Conference of the European Association for Machine Translation (EAMT 2009)*, pages 12–19, 2009.

Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI Global, 2010.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6000–6010. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Lukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. Tensor2tensor for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 193–199, Boston, MA, March 2018. Association for Machine Translation in the Americas. URL http://www.aclweb.org/anthology/W18-1819.

Ekaterina Vylomova, Laura Rimell, Trevor Cohn, and Timothy Baldwin. Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1671–1682, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1158. URL https://www.aclweb.org/anthology/P16-1158.

Alex Waibel, Ajay N Jain, Arthur E McNair, Hiroaki Saito, Alexander G Hauptmann, and Joe Tebelskis. Janus: a speech-to-speech translation system using connectionist and symbolic processing strategies. In *[Proceedings] ICASSP 91: 1991 International Conference on Acoustics, Speech, and Signal Processing*, pages 793–796. IEEE, 1991.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. *Conference on Computer Vision and Pattern Recognition 2019*, 2019.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016.

Hua Wu and Haifeng Wang. Pivot language approach for phrase-based statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 856–863, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P07-1108.

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.

X. Yang and W. Liang. An n-gram-and-wikipedia joint approach to natural language identification. In *2010 4th International Universal Communication Symposium*, pages 332–339, Oct 2010. doi: 10.1109/IUCS.2010.5666010.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

Michal Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. The united nations parallel corpus v1. 0. In *LREC*, 2016.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

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, Austin, Texas, November 2016. Association for Computational Linguistics. URL https://aclweb.org/anthology/D16-1163.

List of Publications

Ondřej Bojar, Ondřej Dušek, Tom Kocmi, Jindřich Libovický, Michal Novák, Martin Popel, Roman Sudarikov, and Dušan Variš. CzEng 1.6: Enlarged Czech-English Parallel Corpus with Processing Tools Dockered. In *International Conference on Text, Speech, and Dialogue*, pages 231–238. Springer, 2016a. ISBN 978-3-319-45509-9.

Ondrej Bojar, Roman Sudarikov, Tom Kocmi, Jindrich Helcl, and Ondrej Cıfka. UFAL Submissions to the IWSLT 2016 MT Track. *IWSLT. Seattle, WA*, 2016b.

Ondřej Bojar, Jindřich Helcl, Tom Kocmi, Jindřich Libovický, and Tomáš Musil. Results of the WMT17 Neural MT Training Task. In *Proceedings of the 2nd Conference on Machine Translation (WMT)*, Copenhagen, Denmark, September 2017.

Jindřich Helcl, Jindřich Libovickỳ, Tom Kocmi, Tomáš Musil, Ondřej Cífka, Dusan Varis, and Ondřej Bojar. Neural Monkey: The Current State and Beyond. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 168–176, 2018.

Tom Kocmi and Ondřej Bojar. SubGram: Extending Skip-Gram Word Representation with Substrings. In *International Conference on Text, Speech, and Dialogue*, pages 182–189. Springer, 2016.

Tom Kocmi and Ondřej Bojar. Curriculum Learning and Minibatch Bucketing in Neural Machine Translation. In *Recent Advances in Natural Language Processing 2017*, September 2017a.

Tom Kocmi and Ondřej Bojar. LanideNN: Multilingual Language Identification on Character Window. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 1: Long papers, pages 927–936. Association for Computational Linguistics, 2017b. ISBN 978-1-945626-34-0.

Tom Kocmi and Ondřej Bojar. An Exploration of Word Embedding Initialization in Deep-Learning Tasks. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, pages 56–64, Kolkata, India, December 2017c. NLP Association of India. URL http://www.aclweb.org/anthology/W/W17/W17-7508.

Tom Kocmi and Ondřej Bojar. Trivial Transfer Learning for Low-Resource Neural Machine Translation. In *Proceedings of the 3rd Conference on Machine Translation (WMT): Research Papers*, Brussels, Belgium, November 2018.

Tom Kocmi and Ondřej Bojar. Transfer Learning across Languages from Someone Else's NMT Model. 2019a. URL https://arxiv.org/abs/1909.10955.

Tom Kocmi and Ondřej Bojar. CUNI Submission for Low-Resource Languages in WMT News 2019. In *Proceedings of the Fourth Conference on Machine Translation, Volume 2: Shared Task Papers*, Florence, Italy, August 2019b. Association for Computational Linguistics.

Tom Kocmi, Dušan Variš, and Ondřej Bojar. CUNI NMT System for WAT 2017 Translation Tasks. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2017)*, 2017.

Tom Kocmi, Shantipriya Parida, and Ondřej Bojar. CUNI NMT system for WAT 2018 Translation Tasks. In *Proceedings of the 5th Workshop on Asian Translation (WAT2018)*, Hong Kong, China, 2018a. Asian Federation of Natural Language Processing.

Tom Kocmi, Roman Sudarikov, and Ondřej Bojar. CUNI Submissions in WMT18. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 435–441, Belgium, Brussels, October 2018b. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6416.

Tom Kocmi, Dušan Variš, and Ondrej Bojar. CUNI Basque-to-English Submission in IWSLT18. In *International Workshop on Spoken Language Translation*, 2018c.

Milan Straka, Nikita Mediankin, Tom Kocmi, Zdeněk Žabokrtskỳ, Vojtěch Hudeček, and Jan Hajic. SumeCzech: Large Czech News-Based Summarization Dataset. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, 2018.

Roman Sudarikov, David Mareček, Tom Kocmi, Dušan Variš, and Ondřej Bojar. CUNI Submission in WMT17: Chimera Goes Neural. In *Proceedings of the 2nd Conference on Machine Translation (WMT)*, Copenhagen, Denmark, September 2017.

List of Abbreviations

BiRNN Bidirectional Recurrent Neural Network

BP Brevity Penalty

BPE Byte Pair Encoding

CNN Convolutional Neural Network

IWSLT International Workshop on Spoken Language Translation

LanideNN Language Identification by Neural Networks

LM Language Model

LSTM Long Short-Term Memory cells

MT Machine Translation

NLP Natural Language Processing

NMT Neural Machine Translation

NN Neural Network

PBMT Phrase-Based Machine Translation

POS Part-of-Speech

OOV Out-of-Vocabulary

RNN Recurrent Neural Network

SMT Statistical Machine Translation

T2T Tensor2tensor

WAT Workshop on Asian Translation

WMT Workshop on Statistical Machine Translation

List of Observations

List of Figures

List of Tables