UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Taido Purason

# Efficient Use of Pre-trained NMT Models Through Mixing and Matching

Master's Thesis (30 ECTS)

Supervisor(s): Andre Tättar, MSc

Tartu 2023

## Efficient Use of Pre-trained NMT Models Through Mixing and Matching

**Abstract:** With an increasing amount of pre-trained language models and neural machine translation (NMT) models becoming available, it is important to investigate how to use them when training new models to avoid expensive training from scratch. This thesis investigates how to effectively use pre-trained models, focusing on combining encoders and decoders of different independent pre-trained NMT models as modules. This is not directly possible since the intermediate representations of any two independent NMT models are different and cannot be combined without modification. To get around this, firstly, a dimension adapter is added if the encoder and decoder have different embedding dimensionalities, and secondly, extra encoder layers are added after the pre-trained encoder to align the intermediate representations. As a proof of concept, this thesis looks at many-to-Estonian translation and combines a massively multilingual encoder and a high-quality language-specific decoder. The results show significant improvements in both translation quality and speed for many-to-one translation over the baseline multilingual model. Furthermore, the ability to rapidly train a high-quality NMT system is successfully demonstrated with Estonain-Ukrainian and Ukrainian-Estonian translation, achieving competitive results compared to previous works. More broadly, the thesis demonstrates that sentence representations of two independent NMT models can be made compatible without changing the pre-trained components while keeping translation quality from deteriorating.

**Keywords:**
natural language processing, neural machine translation, machine translation, multilingual machine translation, artificial neural networks

**CERCS:** P176, Artificial intelligence

## Eeltreenitud neuromasintõlke mudelite efektiivne kasutamine

**Lühikokkuvõte:**

Kuna järjest enam eeltreenitud keelemudeleid ja neuromasintõlke (NMT) mudeleid on vabalt kättesaadavad, on oluline uurida, kuidas neid kasutada uute mudelite treenimisel, et vältida ressursimahukat nullist treenimist. Käesolev töö uurib, kuidas eeltreenitud mudeleid tõhusalt kasutada, keskendudes erinevate eeltreenitud NMT mudelite komponentide kombineerimisele. Erinevate mudelite komponente pole otseselt ilma muutmata võimalik kombineerida, kuna kahe sõltumatu NMT mudeli lausete vektoresitused ei ole ühilduvad. Töös pakutakse välja meetod, kus esmalt lisatakse eeltreenitud kodeerijale mõõtme adapter, kui kodeerija ja dekodeerija omavad erinevaid vektoresituse mõõtmeid, ning seejärel täiendavaid kodeerijakihte. Meetodi demonstreerimiseks kombineeritakse 202 keelt toetava mitmekeelse NMT mudeli kodeerija ja eesti keele spetsiifilise NMT mudeli dekodeerija ning vaadeldakse tõlget eesti keelde. Tulemused näitavad olulist võitu nii tõlkekvaliteedis kui -kiiruses võrreldes mitmekeelse baasmudeliga. Lisaks demonstreeritakse edukalt võimekust treenida kiiresti kvaliteetne NMT süsteem eesti-ukraina ja ukraina-eesti tõlke jaoks, saavutades konkurentsivõimelisi tulemusi võrreldes varasemate töödega. Üldisemalt näitab lõputöö, et kahe sõltumatu NMT mudeli komponendid saab muuta ühilduvaks, sealjuures muutmata eeltreenitud parameetreid ja ohverdamata tõlkekvaliteeti.

**Võtmesõnad:**
loomulikku keele töötlus, neuromasintõlge, masintõlge, mitmekeelne masintõlge, tehisnärvivõrgud

**CERCS:** P176, Tehisintellekt

# Contents

# 1 Introduction

The field of natural language processing (NLP) has witnessed a significant transformation in recent years and has seen an increase in the availability of pre-trained neural machine translation (NMT) models and language models (LMs), which are increasingly large and trained on vast amounts of data. Instead of starting from scratch every time a new and improved system needs to be trained, it makes sense to leverage pre-trained components, both from an economic and environmental perspective.

Previous research has investigated efficient fine-tuning of NMT models (Bapna and Firat, 2019; Philip et al., 2020) and making use of language models in training NMT models (Zhu et al., 2020; Rothe, Narayan, and Severyn, 2020; Chen et al., 2021; Sun, Wang, and Li, 2021; Chen et al., 2022). This thesis proposes and investigates combining components of pre-trained NMT models to efficiently create new NMT models which improve on the previous models by adding new directions, offering better translation quality and speed. In the process, this work also explores how to align the intermediate representations of two independent NMT models so that the sentence representations outputted by the encoder of one model can be decoded by the decoder of another. We mainly look at the case of combining a massively multilingual model's encoder with a language-specific decoder to achieve fast and high-quality machine translation. We hypothesize that the translation quality can be increased since we add a language-specific decoder to a massively multilingual model, which likely suffers from capacity bottleneck as shown to be the case for universal models by previous research (Johnson et al., 2017; Tan et al., 2019; Arivazhagan et al., 2019). Achieving high-quality translation to one language likely requires fewer parameters than in the multilingual case, and thus, we can decrease the total size of the model. We also demonstrate that aligning the representations with data from a few languages can transfer to a much larger set of languages that are unseen by the original decoder and also not in the current training set.

In order to combine an encoder from one NMT model and a decoder from another, there are two obstacles that have to be solved. Firstly, the incompatible representational spaces of the two models need to be aligned, and secondly, the models could have different embedding dimensionalities, which would have to be transformed as well. The approach proposed in this thesis solves both of these issues. The proposed architecture, as illustrated in Figure 1, involves the incorporation of a dimension adapter – a feed-forward layer (linear transformation) – which serves the purpose of converting the dimensionality of the encoder if it differs from that of the decoder. Moreover, we incorporate randomly initialized transformer layers to convert the pre-trained encoder's representation to a representation understandable to the decoder. We freeze the parameters of the original pre-trained models so they remain unchanged, however we also investigate the possibility of unfreezing the original modules to enable further improvements.

To demonstrate our approach, we combine the encoder NLLB (NLLB Team et al., 2022), a massively multilingual NMT model, and the language-specific decoder

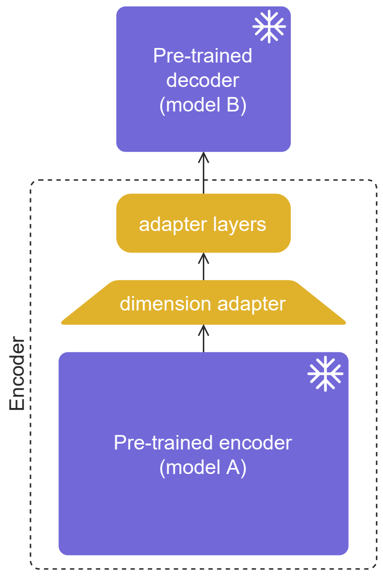

Figure 1. The proposed architecture. Dimension adapter denotes a component that takes input with the dimensionality of model A output and outputs with the dimensionality of model B (for example, a feed-forward layer). Adapter layers are transformer encoder layers. Components from models A and B have frozen parameters.

of MTee (Tättar et al., 2022), an Estonian-centric model covering 4 languages. We focus on improving many-to-Estonian translation directions and demonstrate significant improvements in that regard. The model is more efficient to train than full fine-tuning and also reduces the number of parameters by 40% compared to the NLLB model. This also means reduced running costs of the model in the long term compared to NLLB.

To further show the effectiveness of this approach, we demonstrate it in a real-life scenario – Ukrainian-to-Estonian translation. With the increase of Ukrainian refugees in Estonia, the Ukrainian-Estonian translation direction has become more important. There is, however, a relatively limited amount of Ukrainian-Estonian parallel data. We demonstrate that with our approach, we can produce a high-quality, competitive Ukrainian-Estonian translation system using pre-trained models with only a few hours of training and no Ukrainian-Estonian data required.

We also explored training the decoder from scratch instead of initializing from a pre-trained model and found that this is also a viable approach to creating a new decoder. We demonstrated it by training a Ukrainian decoder for NLLB, which showed competitive translation quality. Since we do not train the parameters of the NLLB encoder, the Estonian and Ukrainian decoders can be combined into a single system without needing multiple large encoders/decoders, making it scalable and simple to deploy.

This thesis provides a comprehensive comparison of the proposed methods to other methods such as pivoting and full fine-tuning. Since most of the available high-quality training data is for English, the resulting systems usually have higher translation quality for English directions. With this in mind, we also evaluate the translation quality with pivoting through English – for example when translating from German to Estonian, we first translate from German to English and then from English to Estonian. We evaluate the scenario with translations to English being created with NLLB and the translation from English to Estonian being created with MTee. For comparison, we also explore a scenario with both translations being produced with NLLB. We confirm that pivoting is a competitive approach, which improves on the baseline NLLB model. However, a major downside of pivoting is its increased translation time since two translations must be produced for one sentence.

The main contributions of this work are the following:

- a novel method to combine components of pre-trained NMT models,
- successful demonstration of the proposed method for many-to-Estonian and many-to-Ukrainian translation,
- a detailed investigation of the proposed methods with comparison to other widely used approaches and previous research,
- an open-source implementation (see Section 4.5).

This thesis is organized into 7 sections including the Introduction (Section 1). Background (Section 2) provides an overview of the core theoretical concepts of NMT model development. Related Works (Section 3) gives an overview of NMT approaches and related works. Approach (Section 4) describes the proposed methods, experiment setup and evaluation. Results (Section 5) reports and analyses the results of this thesis. Discussion (Section 6) provides further discussion relating to the results and future research. Conclusion (Section 7) gives a final overview of the thesis' findings and contributions.

# 2 Background

## 2.1 Transformers

Machine translation is a sequence-to-sequence task usually tackled using encoder-decoder architectures. After being proposed by Vaswani et al., 2017, the Transformer architecture (shown in Figure 2) has become the most widely used architecture in neural machine translation, providing state-of-the-art results over the previously used models.

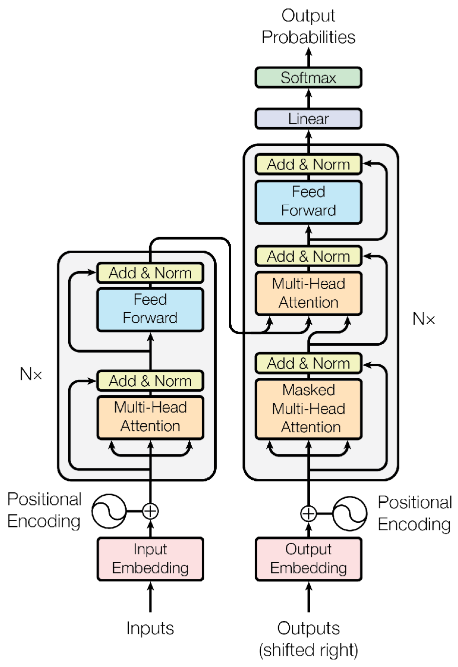

Figure 2. The Transformer architecture (Vaswani et al., 2017).

Vaswani et al., 2017 achieve performance improvements by omitting recurrence and convolutions, commonly used in previous architectures, and instead using only attention mechanisms, implemented as scaled dot-product attention. The authors use two types of attention: self-attention and cross-attention (encoder-decoder attention). Self-attention is present in both the encoder and the decoder, allowing all the positions in the sequence to attend to each other. Cross-attention is present in the decoder, allowing the decoder to attend to the encoder output. The authors also add positional embeddings to encoder and decoder inputs so the transformer has the information on the order of tokens in the sequence.

A major benefit of the transformer architecture over recurrent neural networks (RNN-s) is its parallelizability: it has a constant number of sequential operations as opposed to RNN-s linear operations in relation to the sequence length (Vaswani et al., 2017). The authors also noted that, in sequence modeling tasks such as machine translation, the transformer is desirable because of its ability to model long-term dependencies since it offers O(1) path length between any two dependencies in the sequence. Vaswani et al., 2017 note that one limitation of the transformer is the limited sequence length caused by the quadratic attention complexity in relation to it, making the self-attention more efficient than recurrence only when the sentence length is smaller than the model dimensionality. There are, however, approaches that try to alleviate this issue, such as Dai et al., 2020 and Beltagy, Peters, and Cohan, 2020.

In machine translation, the transformer is usually applied in an auto-regressive manner. The source sentence is processed with the encoder and inputted to the decoder via cross-attention. The decoder then generates the output position by position, using the sequence generated so far as the input (in addition to the encoder output). In other words, a decoder position receives the output at the previous position as an input. Usually, a decoding algorithm is applied to obtain quality output sequences, with one of the most commonly used algorithms being beam search.

Originally, the Transformer was demonstrated on the machine translation task and constituency parsing with an encoder-decoder architecture (Vaswani et al., 2017); however, it has also been applied with encoder-only or decoder-only architectures. For example, BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) are encoder-only language models and GPT-like models, for example, GPT-2 (Radford et al., 2019), and ChatGPT are decoder-only. The Transformer architecture has been applied to other tasks as well, for example, it has been successfully applied in computer vision tasks (Dosovitskiy et al., 2021).

## 2.2 Byte-pair Encoding

Neural machine translation models typically have a finite vocabulary of tokens they can process. Real-world texts rarely have a finite vocabulary of words, raising the need to split the text into tokens. One possible way would be to split the text into individual characters. While being a simple and language-independent solution, this would make the number of tokens in the sequence relatively large, increasing computational costs. Using the most frequent words as a vocabulary has even more issues since we would encounter many rare words that the model has not seen. The vocabulary would also grow very large, which would be computationally ineffective. For example, Estonian is a morphologically rich language, where suffixes would be used instead of prepositions like in English, for example, *"raamatusse"* (*"into a book"*) and *"raamatust"* (*"from a book"*). This would create many variations of the same word root *"raamat"* (*"book"*). Another difficult case would be compound words, for example, *"Donaudampfschifffahrtsgesellschaftskapitän"* in German (translates to *"Danube steamship company captain"*), which is a very rarely used word and not likely to come up in training data; however, the components of that word would likely be common enough for the model to learn to translate it successfully if given them separately. This demonstrates the need for subword segmentation in NMT.

A successful subword segmentation approach was proposed by Sennrich, Haddow, and Birch, 2016, who used the Byte-Pair encoding compression algorithm (Gage, 1994) to learn the segmentation model from the training data. The main idea of the algorithm is that the text is split into characters as subwords and iteratively combined according to the frequency in the training data until the desired vocabulary size is reached. The approach proposed by Sennrich, Haddow, and Birch, 2016 is also language-agnostic and can be learned from the data in an unsupervised manner without requiring any knowledge about the language.

## 2.3 Translation Quality Metrics

Evaluating machine translation models' translation quality is a crucial step in the development process. While human evaluators would provide the most high-quality insight into the model's performance, it is too labor-intensive and expensive to use in the day-to-day development process. This has led to many automatic metrics being used in NMT model development.

### 2.3.1 BLEU

One of the most well-known automatic machine translation evaluation metrics is BLEU, which relies on word n-gram overlap between the candidate and the reference (Papineni et al., 2001). To evaluate a translation it calculates word n-gram precision – the count of n-grams occurring in reference translation divided by total n-grams in the candidate. To penalize too long translations, Papineni et al., 2001 clip the count of each n-gram in the translation by the maximal count in reference. They note that too short translations should also be avoided and thus introduce a corpus-level brevity penalty that penalizes translations that are shorter than the references. They demonstrate that BLEU highly correlates with human evaluations, is language-independent and fast to compute, making it a useful metric for automatic MT evaluation. Typically the n-grams from unigrams to 4-grams are used for calculating BLEU, and the metric values are scaled to range from 0 to 100.

BLEU also has many downsides. For example, since BLEU uses word n-gram precision, it fails to consider synonyms and different variations of the same base word. It is also dependent on how the text is tokenized into words. The correlation of BLEU to human judgments is also lower than other more recent evaluation methods (Freitag et al., 2022).

### 2.3.2 chrF

Character n-gram F-score (chrF), similarly to BLEU, is also a candidate-reference overlap based automatic MT evaluation metric, however, instead of word n-gram precision, it uses character n-gram F-score (Popović, 2015). The authors demonstrated that it outperforms BLEU in human correlation, especially for morphologically rich languages, while having the desirable features of BLEU, such as being language-independent and fast to compute (Popović, 2015; Popović, 2016). chrF's correlation to human evaluations was even further improved by combining it with word n-gram F-score (wordF), yielding chrF+ (up to 6-gram chrF, unigram wordF) and chrF++ (up to 6-gram chrF, bigram wordF) (Popović, 2017). Due to its language independence and relatively high correlation to human judgments, chrF++ is used in this thesis instead of BLEU.

### 2.3.3 Neural Network Based Metrics

There also exist transformer encoder based metrics such as COMET (Rei et al., 2020) and BLEURT (Sellam, Das, and Parikh, 2020). Recently, Kocmi and Federmann, 2023 proposed a GPT-based metric GEMBA, which uses zero-shot prompting with no training or training data required. They have been shown to offer a higher correlation with human evaluation than BLEU and chrF (Freitag et al., 2022). The downsides of these metrics are that they require more resources to compute and need trained data (or a pre-trained model trained with the training data) to support a language, which might not be possible or yield subpar results for low-resource languages. For example, none of the aforementioned metrics support all the languages used in evaluating models in this thesis.

# 3 Related Works

## 3.1 Multilingual Neural Machine Translation

Using a system of single models (one model per translation direction) has been a standard implementation for multilingual NMT. However, this provides no transfer-learning, meaning that low-resource languages will end up with low translation quality, and zero-shot translation is outright impossible. We also face the issue of quadratically growing number of models in relation to the number of languages, should we require to translate in all directions. An alternative would be pivoting, where we translate through a language that has enough resources with the translated languages to achieve higher translation quality for low-resource directions and make zero-shot translation possible (Habash and Hu, 2009). However, pivoting is slower than using a single model directly since the sentences are translated through multiple models. There is also a possibility of errors being propagated more and information being lost.

A more recent widely used approach for multilingual NMT would be using Universal models, where instead of having a separate model for each language pair, we would have a single universal model to handle all directions (Johnson et al., 2017). A target language token is added to the encoder or the decoder input to indicate which language the model should translate into. This has enabled the successful development of massively multilingual NMT models (Aharoni, Johnson, and Firat, 2019; Arivazhagan et al., 2019; Zhang et al., 2020). With the universal model, the low-resource and zero-shot directions benefit greatly from transfer learning. The universal model is also more scalable for many-to-many translation than a system of single models (Dabre, Chu, and Kunchukuttan, 2020). Unfortunately, universal models often suffer from the negative transfer, with high-resource languages having lower translation quality (outperformed by single models) because the model has to share capacity between all the different directions, also known as a capacity bottleneck (Johnson et al., 2017; Tan et al., 2019; Arivazhagan et al., 2019).

Adding sparsity to the models is a way to achieve the best of both worlds regarding adequate transfer learning and providing enough capacity for high-resource directions. Escolano, Costa-jussà, and Fonollosa, 2019 and Escolano et al., 2021 proposed a simple and effective approach to creating modular models with language-specific encoders and decoders, which are jointly trained. They demonstrated that the modular model outperforms the universal model and is extendable by adding new encoder-decoder modules. Lyu et al., 2020 further demonstrated the feasibility of the modular model from the industry's standpoint.

There can also be models with partially shared parameters. For example, some parameters can be shared universally, while others are language-specific. Liao et al., 2021 demonstrate that the zero-shot capabilities of a model with language-specific encoders and decoders can be improved by sharing some layers universally. The largest models of the M2M-100 series also use sparsity this way - the top layers of the decoder are language-group-specific or language-specific, while the rest of the model is universal (Fan et al., 2020).

The sparsity could also be added inside transformer layer components (sub-layer level). One such example would be adapters. Bapna and Firat, 2019 propose using the adapter, which consists of layer normalization, down-projection (feed-forward layer), up-projection, and a residual connection and is added to the end of each transformer layer. They demonstrated the usefulness of adapters for massively multilingual NMT by pre-training a universal NMT model and training translation direction-specific (bilingual) adapters with the rest of the parameters frozen. The authors achieve translation quality on par with single models for high-resource languages while having a more parameter-efficient model and zero-shot capabilities of the universal model. Philip et al., 2020 propose and demonstrate the effectiveness of language-specific (monolingual) adapters. Zhang et al., 2021 add conditional language-specific routing (CLSR) layers/adapters after transformer sublayers. CLSR adapters learn to route each token to either language-specific or shared feed-forward neural network conditioned on the token representation (Zhang et al., 2021).

A prominent direction in creating massively multilingual models is incorporating sparsely-gated mixture-of-experts (MoE) layers (Shazeer et al., 2017), which were applied to transformer architecture by Lepikhin et al., 2020 so that tokens get routed to different feed-forward neural networks (FFNs) instead of a single FFN being used for all tokens (vanilla transformer). The Switch transformer (Fedus, Zoph, and Shazeer, 2021) scaled the transformer up to a trillion parameters by using MoE sub-layers with every token routed to a single expert. The largest NLLB NMT model uses Mixture-of-Experts for scaling the model to 56 Billion parameters (NLLB Team et al., 2022). Gong, Li, and Genzel, 2021 propose a sparse NMT model which selects layers, attention heads, and FFN parameters conditioned on input languages.

## 3.2 Making Use of Pre-trained Language Models

With many pre-trained language models becoming openly available, research into using pre-trained language models as starting points for NMT model training has become increasingly important. Using pre-trained models can be beneficial to achieve higher translation quality and use fewer computational resources.

One approach to using pre-trained models is pre-training encoder-decoder models on sequence-to-sequence (seq2seq) tasks and then fine-tuning on machine translation. MASS is a pre-trained seq2seq model, which was trained on masked sentence fragment reconstruction task and was shown to improve several downstream tasks (including machine translation) when fine-tuned (Song et al., 2019). Liu et al., 2020 propose mBART - a seq2seq denoising autoencoder model trained on BART task in 25 languages. They show that initializing from mBART for machine translation fine-tuning improves translation quality and provides transfer learning for zero-shot directions.

The other approach would be taking existing encoder-only pre-trained language models and/or decoder-only language models and using them for initializing NMT model parameters. Zhu et al., 2020 incorporate BERT (Devlin et al., 2019) sentence representations via the attention mechanism of the NMT model's encoder and decoder. Rothe, Narayan, and Severyn, 2020 use BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and GPT-2 (Radford et al., 2019) for initializing NMT model parameters. SixT (Chen et al., 2021) and SixT+ (Chen et al., 2022) use XLM-RoBERTa (Conneau et al., 2019) to initialize the encoder of an NMT model. They conduct the training in two stages: first, training the randomly initialized decoder and then training the whole model. Sun, Wang, and Li, 2021 propose Graformer, which is constructed by initializing the encoder from a BERT-like model and the decoder from a GPT-like model and adding extra transformer layers to both the encoder and decoder to connect them.

DeltaLM combines the two aforementioned approaches by initializing a seq2seq encoder-decoder model from InfoXLM (Chi et al., 2021), which is an encoder-only language model, and pre-training on span corruption and translation span corruption tasks (Ma et al., 2021). The authors fine-tuned DeltaLM on machine translation and found that it outperforms strong baselines such as the M2M-100 model. The translation system trained from DeltaLM also achieved the highest ranking in WMT21 machine translation shared task (Yang et al., 2021).

## 3.3 Pre-trained NMT models

There are many pre-trained NMT models already openly available for use. OpusMT provides over 1000 NMT models, most of which are bilingual, but some also multilingual (Tiedemann and Thottingal, 2020). Rothe, Narayan, and Severyn, 2020 published NMT models which were initialized from BERT and trained on the NMT task. M2M-100 is a series of NMT models (varying in size) which were trained on 7.5B sentence pairs and support translation between 100 languages (Fan et al., 2020). The NLLB-200 NMT model further improves it and extends support to 200 languages with a training dataset of 18B sentence pairs (NLLB Team et al., 2022). Both M2M-100 and NLLB-200 are strong baselines in NMT research regarding translation quality. MetaAI has also made their WMT shared news/general translation task NMT models public (Ng et al., 2019; Chen et al., 2020; Tran et al., 2021). MTee provides an Estonian-centric (Estonian to/from English, German, Russian) NMT model with language-specific encoders-decoders (Tättar et al., 2022).

# 4 Approach

## 4.1 Architecture and Models

Our proposed approach enables the combination of any two pre-trained transformer-based NMT model's encoder and decoder without modification. Furthermore, it also allows language models using the transformer encoder architecture to be used as the NMT model's encoder.

To achieve this, we add randomly initialized transformer layers, which we refer to as adapter layers, to the end of the encoder. This allows us to align the encoder and decoder representational space without changing the pre-trained modules by keeping them frozen and only training the added layers. If the encoder's dimensionality differs from the decoder's dimensionality, we add a dimensionality adapter, either before, after, or in-between the adapter layers. The dimensionality adapter is defined as a simple feed-forward linear transformation with learnable parameters $W$ and $b$:

$$DimAdapt(x) = Wx + b$$
$$W \in \mathbb{R}^{d_2 \times d_1}, b \in \mathbb{R}^{d_2}$$

Layers before the dimensionality adapter will have the pre-trained encoder's dimensionality ($d_1$), while the layers following the dimensionality adapter will have the pre-trained decoder's dimensionality ($d_2$).

We add extra layers to the encoder and not decoder to avoid increasing the decoder size and achieve higher computational efficiency in inference with beam search since the encoder representation will be computed once. In contrast, the decoder representation will be computed multiple times depending on the beam size.

Inspired by Chen et al., 2021 and Chen et al., 2022, we also use 2-stage training, where we first train with pre-trained parameters frozen and then continue training with some or all of them unfrozen. In most cases, we unfreeze only the decoder since it is computationally less expensive to train than the encoder in our experiments. We also experiment with 1-stage training, where we only train the randomly initialized parameters and leave pre-trained parameters frozen.

In the main experiments, we use the *NLLB-1B-distilled* model. Unless otherwise mentioned, NLLB or NLLB-1B refers to *NLLB-1B-distilled*. NLLB-1B has 1.3B parameters with 24/24 layers in encoder/decoder, 1024 embed dim., 8192 feed-forward dim., and 16 attention heads. We also use *NLLB-600M-distilled* in Section 5.5.4. For comparison, we also report NLLB MoE 54B parameter model results from NLLB Team et al., 2022 but do not use it in our models because of its large size. We also use *XLM-RoBERTa-large* (Conneau et al., 2019) (355M parameters) to demonstrate that our method works with encoder-based language models.

For the pre-trained decoder, we use the Estonian decoder from MTee (Tättar et al., 2022). MTee has 227M parameters, however, it has language-specific encoders and decoders following the *transformer-base* architecture (6/6 layers encoder/decoder, 512 embed dim., 2048 feed-forward dim., 16 attention heads), so translating in one direction uses 57M parameters.

We add a dimension adapter followed by two adapter layers to the base pre-trained encoder in the main experiments.

We also experiment with using a randomly initialized decoder with an NMT pre-trained encoder and apply the same approach with adapters to use a narrower (smaller dimensionality) and deeper (larger number of layers) decoder instead a wider and shallower one with the same number of parameters following the encoder dimensionality.

## 4.2 Dataset

The **training dataset** for the main experiments is created from the CCMatrix corpus (Schwenk et al., 2019). We use English-Estonian, German-Estonian, French-Estonian, and Polish-Estonian directions (see dataset sizes in Table 1). We additionally have experiments where we investigate many-to-Ukrainian translation. The training set for those experiments is composed of CCMatrix Estonian-Ukrainian and English-Ukrainian data.

Table 1. Main many-to-Estonian and many-to-Ukrainian parallel dataset compositions in the number of sentence pairs (CCMatrix).

| Target | Source | | | | |
|---|---|---|---|---|---|
| | ET | EN | DE | FR | PL |
| ET | - | 22.0M | 12.5M | 11.8M | 7.0M |
| UK | 2.0M | 20.0M | - | - | - |

We use FLORES-200 *dev-test* as our **test set** and *dev* as the **validation set**. We trust that NLLB training set did not contain examples from FLORES-200 *dev-test* and *dev*, since they were used for evaluation. Furthermore, we also confirm that MTee training dataset did not contain these evaluation sets. The translations are evaluated for all 201 FLORES-200 directions with Estonian as the target (many-to-Estonian translation).

In Section 5.5.4, we use Europarl (Tiedemann, 2012) English-Estonian, German-Estonian, French-Estonian, Polish-Estonian, Latvian-Estonian, and Finnish-Estonian directions with the dataset composition reported in Table 2

Appendix II. Table 17 provides the language codes used when abbreviating language names in this thesis.

Table 2. Additional ablation dataset composition in the number of sentence pairs (Europarl).

| Target | Source | | | | | |
|---|---|---|---|---|---|---|
| | EN | DE | FR | PL | FI | LV |
| ET | 651236 | 578248 | 630126 | 639046 | 620939 | 637468 |

## 4.3 Data Preprocessing

The training data is filtered for validation and test set overlaps. We use SentencePiece (SP) BPE (Kudo and Richardson, 2018) segmentation models from the respective pre-trained models. For example, if NLLB is used as an encoder, we preprocess the source sentences with the NLLB SP model, if MTee is used as a decoder we preprocess the target sentences with the MTee SP model. If MTee vocabulary and SentencePiece models are used, we also use the MTee normalization script. An example of the data pre-processing workflow is illustrated by Table 3.

Table 3. Example of data pre-processing workflow: first, the punctuations get normalized, then the normalized text is encoded with SentencePiece (SP). Here we display the whitespace escape symbol SentencePiece uses as an underscore, while it actually uses the Unicode symbol U+2581.

| | |
|---|---|
| ORIGINAL | `Neid platoosid nimetatakse sageli „viddedeks”.` |
| NORMALIZED | `Neid platoosid nimetatakse sageli "viddedeks".` |
| SP ENCODED | `_Neid _pla toos id _nimetatakse _sageli _" vid ded eks " .` |

## 4.4 Evaluation

chrF++[1] (Popović, 2017) is used as the primary metric for translation quality, however, chrF[2] (Popović, 2015) is also reported in some cases for comparison with previous works. The metric values are calculated using the standard SacreBLEU (Post, 2018) implementation for both of the metrics.

COMET scores (Rei et al., 2020) are also provided for a selection of languages in Section 5.1. Specifically, *Unbabel/wmt22-comet-da* model is used (Rei et al., 2022).

The main training experiments are repeated for 5 random seeds and we report confidence intervals ($p = 0.01$, using Student's t-distribution) for the metrics.

[1]sacreBLEU signature: `nrefs:1|case:mixed|eff:yes|nc:6|nw:2|space:no|version:2.3.1`

[2]sacreBLEU signature: `nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.3.1`

Additionally, a modified version of the Win Ratio (WR) is calculated. Win-ratio, as applied to NMT evaluation by Zhang et al., 2020, measures for what proportion of the translation directions a proposed solution outperforms the baseline in terms of a translation quality metric. We incorporate significance testing into the metric and propose the Win Ratio with Significance (WRS), where we report the percentage of directions that the baseline is outperformed with significance ($p = 0.01$, one-sample t-test based on 5 seeds). Additionally, we provide WRS based on Paired Bootstrap Resampling t-test (Koehn, 2004) at $p = 0.01$, calculated for a single seed. For the main experiments, we also report the Win ratio over all models (WR/all), which shows on what percentage of directions the model achieves the highest score over all models. WR/all is used for additional insight and it does not necessarily mean that a given model is significantly better than others.

All translations are acquired using beam search decoding with a beam size of 4. We use the best checkpoint according to validation loss for final evaluations.

## 4.5 Training

Fairseq (Ott et al., 2019) is used for implementing the training and models. The model implementation, along with usage instructions, is made public on GitHub[3].

The models are trained for a total of 100k updates with a learning rate of 0.0005. If 2-stage training is used, the first will be with a learning rate of 0.0005 for 50k updates and the second will be with 0.0001 for 50k updates. Mixed precision (*fp16*) is used during training and inference. Inverse square root learning rate scheduler with 4000 warm-up steps is used in all experiments, including 2-stage. Dropout and attention dropout of 0.1 are used. Adam (Kingma and Ba, 2015) optimizer is used.

The models were trained on 8 GPUs with a batch size of 4096 tokens per GPU (32768 tokens in total). For the main experiments, 4 AMD Instinct MI250X 128GB HBM2e GPUs (acting as 8 GPUs) on LUMI supercomputer were used. Additional training (Many-to-Ukrainian experiments) and evaluation was done using at the University of Tartu HPC Center (University of Tartu, 2018) using NVIDIA Tesla V100-32GB and NVIDIA A100-40GB GPUs. Gradient accumulation is applied to keep the batch size consistent between experiments when less than 8 GPUs are used.

## 4.6 Writing

Grammarly[4] is used to improve the grammatical correctness and readability of this thesis. ChatGPT[5] is also used for improving the readability of the written text in some cases. The

[3]https://github.com/TartuNLP/mix-and-match-nmt

[4]https://www.grammarly.com/

[5]ChatGPT is a large language model based chatbot created by OpenAI and accessible through https://chat.openai.com/.

following prompt is used *"Improve the readability of the following text: <input text>"*, where <*input text*> is the text written by the author (usually a paragraph). The output might then be used to edit the author's text to improve its readability.

# 5 Results

## 5.1 Main Results

Table 4. Many-to-Estonian translation average chrF++ scores. For experiments involving model training, the average of 5 random seeds are reported with confidence intervals ($p = 0.01$). Average chrF++ and WR/all is reported for all directions and official EU languages separately. WR/all (Win ration over all) reports what percentage of directions achieve the maximal score over all models. WRS (Win ratio with significance, $p = 0.01$) reports what percentage of directions outperform the baseline with both significance based on t-test on 5 seeds and significance based on paired bootstrap resampling t-test (PBR). † - Scores reported by NLLB Team et al., 2022.

| | Model | average chrF++ ↑ | | WR/all ↑ | | WRS ↑ | |
|---|---|---|---|---|---|---|---|
| | | full | EU | full | EU | t-test | PBR |
| (1) | NLLB-1B | 40.2 | 46.7 | 0.0 | 0.0 | - | - |
| (2) | NLLB-MoE[†] | 43.0 | 49.6 | 22.9 | 4.3 | - | 99.5 |
| | **Pivot**, m2en: NLLB-1B | | | | | | |
| (3) | en2et NLLB-1B | 41.4 | 47.5 | 0.0 | 0.0 | - | 84.6 |
| (4) | en2et: MTee | **43.4** | 50.2 | **44.3** | 13.0 | - | **100.0** |
| | **Fine-tune** NLLB-1B | | | | | | |
| (5) | - | 42.5 ± 0.1 | 50.1 ± 0.3 | 3.5 | 0.0 | 91.0 | 86.6 |
| (6) | freeze enc | 43.0 ± 0.1 | 50.3 ± 0.2 | 6.5 | 0.0 | **98.0** | 98.5 |
| | **Ours:** NLLB-1B enc + | | | | | | |
| (7) | rand dec | 42.6 ± 0.3 | 50.2 ± 0.3 | 0.0 | 0.0 | 93.5 | 97.5 |
| (8) | MTee dec | 42.5 ± 0.1 | 50.4 ± 0.1 | 0.0 | 0.0 | 92.0 | 89.1 |
| (9) | MTee dec, 2-stage | 43.1 ± 0.1 | **50.9 ± 0.1** | 26.4 | **87.0** | 93.0 | 96.5 |

The main results are reported in Table 4 with the number of parameters and training times of the models being reported in Table 5. NLLB-1B-distilled (referred to as NLLB-1B) is used as a baseline. Additionally, results of the largest publicly available NLLB model (NLLB-MoE) with 54.5B parameters reported by (NLLB Team et al., 2022) are used for comparison. The table lists average chrF++ scores over all many-to-Estonian translation directions and all official EU languages[6]. The EU language averages are

[6]Bulgarian, Croatian, Czech, Danish, Dutch, English, Estonian, Finnish, French, German, Greek, Hungarian, Irish, Italian, Latvian, Lithuanian, Maltese, Polish, Portuguese, Romanian, Slovak, Slovenian, Spanish, and Swedish

reported to highlight the translation quality for languages more closely related to Estonian and also more frequently translated from. We analyze the quantitative results of pivoting, fine-tuning, and our mixing and matching approach of combining the encoder and the decoder of different pre-trained models.

Table 5. Parameters and training times of many-to-Estonian translation models reported in Table 4. For 2-stage training, we report the parameters of both stages delimited by /. For pivoting, the number of parameters takes into account two passes through the translation model(s).

| | Model | Parameters | | Train. time |
|---|---|---|---|---|
| | | train | total | (hrs.) |
| (1) | NLLB-1B | - | 1.37B | - |
| (2) | NLLB-MoE | - | 54.5B | - |
| | **Pivot**, m2en: NLLB-1B | | | |
| (3) | en2et NLLB-1B | - | 2.74B | - |
| (4) | en2et: MTee | - | 1.45B | - |
| | **Fine-tune** NLLB-1B | | | |
| (5) | - | 1.37B | 1.37B | 22.3 |
| (6) | freeze enc | 604M | 1.37B | 15.0 |
| | **Ours:** NLLB-1B enc + | | | |
| (7) | rand dec | 51M | 817M | 4.4 |
| (8) | MTee dec | 13M | 817M | 3.9 |
| (9) | MTee dec, 2-stage | 13M/51M | 817M | 4.1 |

### 5.1.1 Pivoting

Pivoting is one of the simplest approaches to extending models with new translation directions. Since English is often the most abundant language in training sets, we use it as a pivot language. First, we pivot through English with only the NLLB-1B model, meaning that we first translate to English and then from English to Estonian with the same model. Even though NLLB was trained multilingually and with data balanced to represent non-English languages, translating through English provides better results than directly translating between two languages. NLLB-1B English pivoting for many-to-Estonian translation results in an average 1.2 chrF++ point improvement across all directions, as reported in Table 4 (3). It significantly outperforms the baseline NLLB-1B model on 84.6% of directions according to the PBR t-test. Additionally, when NLLB-1B is used to translate to English and MTee is used for English to Estonian translation ((4) in Table 4), the translation quality is improved by 3.2 chrF++ points compared to the baseline (1), achieving a chrF++ score comparable to the largest NLLB model, which utilizes significantly more parameters. This approach significantly outperforms the baseline on all translation directions, according to the PBR t-test. These results demonstrate that pivoting can enhance translation quality without additional training. However, pivoting requires passing through two models, which increases the time required for translation and reduces long-term cost efficiency.

### 5.1.2 Fine-tuning

Fine-tuning is most commonly applied for adapting pre-trained models to a given translation direction. In our research, we experimented with two different fine-tuning strategies: full fine-tuning (5) and fine-tuning only the decoder of the baseline NLLB model with the encoder frozen (6). We found that both approaches lead to significant improvements over the baseline: 2.3 and 2.8 chrF++ points, respectively. However, the fine-tuning method that involved a frozen encoder was faster to train and yielded a higher chrF++ score on average. Moreover, this method exhibited superior performance compared to the baseline across more language pairs, as confirmed by the t-test WRS scores: 98.0% for the frozen encoder method vs. 91.0% for full fine-tuning. Overall, the findings suggest that the fine-tuning technique with a frozen encoder could effectively and efficiently improve pre-trained NMT models.

### 5.1.3 Mixing and Matching

We look at combining the NLLB-1B encoder and MTee decoder with adapter layers. It can be observed that the NLLB enc + MTee dec model (8), which only trains the adapter (13M parameters) and freezes the pre-trained components, outperforms the baseline in 92.0% of the directions with $p = 0.01$ according to the t-test (89.1% according to PBR), with an average improvement of 2.3 chrF++ points. Among the trained models, the two-stage training approach (9) – training the adapter first (13M parameters), followed by training the adapter with the decoder (51M parameters) – achieved the best results. This method outperforms the baseline by 2.9 chrF++ points on average across all directions and achieves similar average chrF++ scores to the 54B parameter NLLB model. It is only slightly behind the best-performing pivoting model in terms of average chrF++ scores. Additionally, we observed that the two-stage training approach significantly (with $p = 0.01$) outperforms the baseline on 93% of the language pairs according to the t-test (96.5% according to the PBR). However, the fine-tuning method with a frozen encoder showed significant improvements over the baseline in 5% more directions than our approach, indicating room for further improvement. It also achieves the highest score on 26.4% of language pairs compared to all the other models without taking significance into account (WR/all).

We also evaluated a decoder that was randomly initialized with the same architecture (including vocabulary and input/output embeddings) as MTee (experiment (7)), and trained it in a single stage with a frozen encoder, only training the adapter and decoder. It outperformed the baseline by 2.4 chrF++ points on average. We observed that this method performs similarly to the initialized model with no decoder training. Although it is still slightly outperformed by the two-stage model with the pre-initialized decoder in terms of the average chrF++ score, it can be useful when a high-quality pre-trained model for the decoder is not available.

When we look at the EU languages, our best model achieves the highest scores for 87.0% of these languages. For EU languages, NLLB-enc+MTee-dec, 2-stage (9) achieves the highest average chrF++ score and outperforms the baseline by 4.2 chrF++ points. This shows that our method achieves the best result for more closely related languages, whereas for more distant languages, the pivoting approach of combining two models was better. This could be because our training data was composed of EU languages. Furthermore, the pre-trained decoder was also trained on two EU languages and Russian, which could contribute to the high performance on EU languages.

Table 6. Many-to-Estonian translation chrF++ scores for selected directions. Confidence intervals are based on 5 random seeds. † - Scores reported by NLLB Team et al. (2022). Language abbreviations following NLLB Team et al. (2022).

| Model | eng_Latn | deu_Latn | rus_Cyrl | zho_Hans | arb_Arab |
|---|---|---|---|---|---|
| NLLB-1B | 52.6 | 48.5 | 46.6 | 40.2 | 45.8 |
| NLLB-MoE† | 56.1 | 51.8 | 49.5 | 43.8 | 49.1 |
| MTee | 56.9 | 52.2 | 49.9 | - | - |
| **Pivot**, m2en: NLLB-1B | | | | | |
| en2et NLLB-1B | 52.6 | 48.7 | 47.2 | 42.4 | 46.8 |
| en2et: MTee | 56.9 | 52.4 | 49.8 | **45.5** | **49.5** |
| **Fine-tune** NLLB-1B | | | | | |
| - | 56.6 ± 0.3 | 52.3 ± 0.5 | 50.1 ± 0.2 | 44.5 ± 0.2 | 48.8 ± 0.2 |
| freeze enc | 56.2 ± 0.4 | 52.3 ± 0.3 | 50.1 ± 0.2 | 44.6 ± 0.2 | 48.8 ± 0.2 |
| **Ours:** NLLB-1B enc + | | | | | |
| rand dec | 56.1 ± 0.4 | 52.0 ± 0.5 | 49.8 ± 0.5 | 44.1 ± 0.3 | 48.6 ± 0.3 |
| MTee dec | 56.7 ± 0.5 | 52.4 ± 0.4 | 49.9 ± 0.3 | 43.5 ± 0.3 | 48.6 ± 0.2 |
| MTee dec 2-stage | **57.3 ± 0.3** | **52.8 ± 0.2** | **50.4 ± 0.3** | 44.6 ± 0.4 | 49.1 ± 0.3 |

In Table 6, we present the chrF++ scores for translations from a selection of languages to Estonian, serving as an example. It also shows the comparison with the MTee model for the languages supported by the pre-trained MTee model. The mix-and-match models (ours) achieve similar performance to the MTee model, with the two-stage model outperforming it slightly. It can also be seen that for Chinese and Arabic, our approach is outperformed by pivoting with NLLB and MTee. Full chrF++ evaluation results for the best-performing model (NLLB enc + MTee dec 2-stage) and the baseline are available in Appendix I. Table 16.

Table 7. Many-to-Estonian translation COMET scores for selected directions. Underlined results indicate a significant gain over the baseline NLLB-1B with $p = 0.01$ according to Paired Bootstrap Resampling t-test. † - Scores calculated from translations reported by NLLB Team et al. (2022). Language abbreviations are following NLLB Team et al. (2022).

| Model | eng_Latn | deu_Latn | rus_Cyrl | zho_Hans | arb_Arab |
|---|---|---|---|---|---|
| NLLB-1B | 0.8967 | 0.8805 | 0.8700 | 0.8435 | 0.8492 |
| NLLB-MoE[†] | **0.9144** | **0.9031** | **0.8904** | **0.8826** | **0.8781** |
| MTee | 0.8916 | 0.8908 | 0.8819 | - | - |
| **Pivot**, m2en NLLB-1B | | | | | |
| en2et NLLB-1B | 0.8967 | 0.8808 | 0.8705 | 0.8673 | 0.8583 |
| en2et MTee | 0.8916 | 0.8899 | 0.8782 | 0.8788 | 0.8615 |
| **Fine-tune** NLLB-1B | | | | | |
| - | 0.8954 | 0.8878 | 0.8825 | 0.8775 | 0.8631 |
| freeze enc | 0.8974 | 0.8912 | 0.8812 | 0.8772 | 0.8552 |
| **Ours:** NLLB-1B enc + | | | | | |
| rand dec | 0.9001 | 0.8902 | 0.8793 | 0.8688 | 0.8561 |
| MTee dec | 0.9049 | 0.8953 | 0.8831 | 0.8659 | 0.8586 |
| MTee dec 2-stage | 0.9060 | 0.8929 | 0.8857 | 0.8724 | 0.8607 |

We also provide the COMET scores for the same set selected of directions in Table 7 to provide a more reliable metric to evaluate the results. These support the same conclusions with a few exceptions. Firstly, it can be seen that the NLLB-MoE model is always the best-performing model, whereas, with the chrF++ evaluation, it was sometimes outperformed. This casts doubt on whether any of the proposed models outperform NLLB-MoE. However, these results still show the proposed methods' superiority over

NLLB-1B. We also see less improvement in English scores over the baseline NLLB model compared to the chrF++ results – only the mix-and-match models with the MTee decoder outperform it significantly.

#### 5.1.4 Efficiency

NLLB-1B enc. + MTee dec. reduces the number of parameters by 40% compared to the baseline model and the default fine-tuning approach (see Table 5). Even though we add 13M trainable parameters to the encoder (adapter layers), we use a significantly smaller decoder than NLLB-1B, leading to fewer trained and total parameters. This makes the training time of our method (NLLB-enc+MTee-dec, 2-stage, 4.1 hours) 5.4 times faster than the full fine-tuning (22.3 hours). We also found that inference with NLLB-enc+MTee-dec is approximately 6.5 times faster than with NLLB-1B (and methods fine-tuning it). This implies that our method's translation speed gains are even larger relative to the pivoting methods that are using NLLB-1B. This demonstrates that our approach offers a more efficient and cost-effective alternative that delivers comparable or better translation quality, with the added benefit of faster training, fewer parameters, and faster inference.

## 5.2 Estonian-to-many Translation

To test the Estonian-to-Many capability of this method, we used the pre-trained MTee encoder as the encoder, pre-trained NLLB-1B as the decoder, and trained only the adapter. This yielded an average chrF++ score of 33.5 – a result 4 points worse than the NLLB-1B baseline model (37.5 chrF++ points). It suggests that adapting a smaller monolingual encoder to a large multilingual does not yield good results. It is possible that this would be more feasible with a larger, more multilingual encoder, more training data and/or further training of pre-trained components, and we leave it for future research to investigate. This leads us to suggest creating a system of a strong multilingual encoder and multiple language-specific decoders with a mix-and-match method, similar to how Chen et al., 2022 extend their approach to multiple languages. This would yield a many-to-few system with modular language-specific decoders. We explore this approach further by creating a Ukrainian decoder in Section 5.4.

## 5.3 Ukrainian-Estonian Translation

Recent events have created an increased need for Ukrainian-Estonian machine translation, with many Ukrainian refugees arriving in Estonia. We demonstrate that our method can be used to rapidly develop competitive NMT models with limited or no data, which could be important in crises. As a comparison, we use previous work by Bergmanis and Pinnis, 2022, and for compatibility with them, we report chrF scores. Additionally,

Table 8. Ukrainian (Cyrillic) to Estonian (Latin) translation chrF scores on FLORES-101 *devtest*. NLLB-1B model was used for all experiments, except for NLLB-MoE (54B). † - reported by Bergmanis and Pinnis, 2022. ‡ - calculated from translations reported by NLLB Team et al., 2022.

| Model | chrF ↑ |
|---|---|
| NLLB-1B | 50.9 |
| NLLB-MoE‡ | 54.0 |
| NLLB-MTee EN pivot | 54.5 |
| NLLB-enc+MTee-dec | 54.6 ± 0.2 |
| NLLB-enc+MTee-dec, 2-stage | 55.0 ± 0.1 |
| Bergmanis and Pinnis (2022)† | 53.5 |
| e-translate† | 53.6 |
| Google† | 56.2 |

we also compare our system to online machine translation system scores reported by them. Our best models from the main experiments (NLLB-enc+MTee-dec models) outperform the system trained by Bergmanis and Pinnis, 2022 judging by chrF scores (see Table 8). Our models also outperform NLLB models – both the baseline NLLB-1B, the NLLB-MoE model – and also the pivoting approach between NLLB and MTee. When compared to public systems, our method outperforms e-translate, however, it underperforms Google Translate (both reported by Bergmanis and Pinnis, 2022). We can also see that the two-stage training method only slightly outperforms the single-stage model, which leaves the pre-trained components unchanged. These results demonstrate that even without Ukrainian data in our training set, we can develop competitive NMT models. This method could also be used in future crises when it is necessary to achieve high-quality NMT without having time to gather significant resources for training from scratch. It should be noted, however, that this comparison is limited by a single test set and automatic evaluation.

## 5.4 Estonian-Ukrainian Translation

We also tried training a Ukrainian decoder for the NLLB encoder, first with Estonian-Ukrainian CCMatrix subset consisting of 2M sentences and secondly additionally including English-Ukrainian CCMatrix data (20M sentences). The Ukrainian decoder consists of 6 layers (transformer-base, i.e. 512 embedding dim. / 2048 feed-forward dim.). We additionally add the dimension adapter and 4 adapter layers in the encoder like in previous experiments. During training the NLLB encoder remains frozen. We compare our results to Bergmanis and Pinnis, 2022 similar to the previous section.

Training with only the 2M Estonian-Ukrainian data, our model (NLLB-enc + rand-dec (ET-UK)) achieves 49.1 chrF points which is slightly more than the pre-trained NLLB-1B model's 48.7 (see Table 9), however, it still uses fewer parameters. The score is improved by 1.7 chrF points when we additionally use English-Ukrainian CCMatrix data for training (NLLB-enc + rand-dec (ET,EN-UK)). This demonstrates a way to use available multilingual data - we see that the positive effect of additional 20M English-Ukrainian sentences transfers to Estonian-Ukrainian translation quality. The model (NLLB-enc + rand-dec ET,EN-UK) also outperforms the baseline model reported by Bergmanis and Pinnis, 2022, however, it is outperformed by NLLB-MoE model.

Table 9. chrF scores for translation into Ukrainian. † - reported by Bergmanis and Pinnis, 2022. ‡ - calculated from translations reported by NLLB Team et al., 2022. <u>Underlined</u> scores are significantly higher than the NLLB-1B baseline according to the Paired Bootstrap Resampling test with $p = 0.01$. Note that scores reported by Bergmanis and Pinnis (2022) are not tested for significance.

| Model | chrF ↑ | | |
|---|---|---|---|
| | ET-UK | LV-UK | LT-UK |
| NLLB-1B | 48.7 | 48.7 | 47.8 |
| NLLB-MoE‡ | <u>51.3</u> | <u>51.5</u> | <u>50.7</u> |
| single ET-UK model | 46.2 | - | - |
| universal ET,EN-UK model | 46.6 | - | - |
| NLLB-enc + rand-dec (ET-UK) | 49.1 | 49.1 | 47.5 |
| NLLB-enc + rand-dec (ET,EN-UK) | <u>50.8</u> | <u>51</u> | <u>49.5</u> |
| Bergmanis and Pinnis (2022) baseline† | 49.5 | 47.6 | 49.4 |
| Bergmanis and Pinnis (2022) BT† | - | - | 50 |
| e-translate† | 50.6 | 53 | 49.1 |
| Google† | 52.4 | 51.4 | 50.8 |

We also train a single-directional transformer model with Estonian-Ukrainian data, and a universal transformer model with English-Ukrainian and Estonian-Ukrainian data to investigate how much the pre-training of the encoder helped. Both follow the transformer base architecture (6/6 encoder/decoder layers, 512 embed. dim., 2048 feed-forward dim., 16 attention heads). We see that using English-Ukrainian data has helped improve the translation quality compared to only Estonian-Ukrainian data. It can also be seen that both are outperformed by the models that initialize the encoder from NLLB which leads us to conclude that the pre-trained encoder offers a significant benefit to the translation quality. It is possible that the results of the single-directional and universal models trained from scratch could be improved, given more exploration of hyperparameters and additional filtering of data.

Since the new Ukrainian decoder can translate from all FLORES-200 languages thanks to the multilingual encoder, we also evaluate the translation quality for other Baltic languages. We can see that our models are quite competitive even though we did not use any Latvian or Lithuanian data for training. For Latvian-Ukrainian, our best-performing model (NLLB-enc + rand-dec ET,EN-UK) significantly outperforms NLLB-1B. It also outperforms the model trained by Bergmanis and Pinnis, 2022 by 3.4 chrF points. For Lithuanian-Ukrainian NLLB-enc + rand-dec ET,EN-UK significantly outperforms NLLB-1B but achieves similar results to Bergmanis and Pinnis, 2022 baseline system (which scored 0.1 points lower) and is outperformed by their system using back-translated data by 0.5 points.

When looking at averages over all FLORES-200 directions (see Table 10 we find that using the limited training data of Estonian-Ukrainian is insufficient to improve the NLLB-1B over most directions, yielding 0.5 chrF++ points lower scores on average. However, when also using the English-Ukrainian data, we improve on the baseline NLLB model by 1.2 chrF++ points. We also see similar improvements when looking at the official EU language averages.

Table 10. Many-to-Ukrainian translation average chrF++ scores on FLORES-200 (201 directions).

| Model | chrF++ ↑ | |
|---|---|---|
| | full | EU |
| NLLB-1B | 39.4 | 46.3 |
| NLLB-enc + rand-dec (ET-UK) | 38.9 | 46.0 |
| NLLB-enc + rand-dec (ET,EN-UK) | 40.6 | 48.1 |

This demonstrates the training of a new decoder can be a high-performing and scalable approach for most cases since with the current configuration, each additional language would increase the model size by about 51M parameters.

## 5.5 Ablation

### 5.5.1 Multi-stage Training

The main results section already discussed training only the adapter and training in 2 stages: first, the adapter, then the adapter, and the decoder. Here we explore additional strategies, both single-stage, and multi-stage (see Table 11). It can be seen that for both single and 2-stage training, training configurations involving training the decoder yield the best results. It is also apparent that training the encoder takes longer due to the higher number of trained parameters and does not provide any benefits. This could be because our training dataset includes four languages, however, we evaluate over 201 directions. Hence, the encoder could improve in these four languages and forget more distant ones. It should be noted that encoder training might lead to different conclusions when the training and test sets are from a different domain compared to the pre-trained encoder. The best scoring model is achieved through a 2-stage training approach that first trains the adapter and then the adapter and decoder. It is also the second-fastest after the single-stage model that only trains the adapter. It can also be noted that initializing the decoder randomly achieves a slightly worse score when compared to a pre-trained decoder. However, this difference might not be significant.

Table 11. Comparison of training strategies' many-to-Estonian translation chrF++ scores. All models listed have 817M total parameters. Trained parameters are based on the last stage and models follow the NLLB-1B+MTee mix-and-match model structure. The stage column describes which parameters are trained. A - dim. adapter and adapter layers, D - decoder, E - encoder.

| Training setup | | | Trained | Time | chrF++ |
|---|---|---|---|---|---|
| dec. init. | stage/trained modules | | params | (hrs) | avg |
| | single | | | | |
| random | A+D | | 51M | 4.3 | 42.8 |
| MTee | A+D | | 51M | 4.4 | 42.9 |
| MTee | A | | 13M | 3.8 | 42.4 |
| | I | II | | | |
| random | A+D | E+A+D | 817M | 5.5 | 42.7 |
| MTee | A | A+D | 51M | 4.0 | 43.2 |
| MTee | A | E+A | 779M | 7.5 | 42.1 |
| MTee | A | E+A+D | 817M | 7.2 | 42.8 |

### 5.5.2 The Effect of Adapter Layers

We explore the effect that using the adapter layers has on model training. We use the NLLB-1B encoder and initialize the decoder randomly to only look at the effect of using a decoder with smaller dimensionality (enabled by the adapter) instead of using a wider decoder following the pre-trained encoder's architecture (without using the adapter). In Table 12, it can be seen that training the model with a 6-layer decoder with a narrow (smaller dimensionality) architecture leads to the best performance with the least training time out of the explored models (MTee arch). It can be seen that the NLLB arch. (6) – a model with six randomly initialized decoder layers, the same number as the MTee arch. model – has over three times more trainable parameters, takes over 70% longer to train, and achieves a slightly lower chrF++ score on average. NLLB architecture model (NLLB arch (2)) with roughly the same number of trained parameters (decoder parameters) as the MTee arch (achieved by using two decoder layers) achieves 4.6 chrF++ points lower translation quality score on average. These results suggest that using the adapter, even without pre-trained decoder, can speed up training and inference with achieving the same or better translation quality. However, conclusively claiming that it provides a better translation quality requires more investigation into the effects of the added layers in the encoder, the used vocabulary, and training hyperparameters.

Table 12. Comparison of models with a **randomly initialized decoder and frozen NLLB-1B encoder**. Average many-to-Estonian chrF++ is reported. *MTee arch.* - encoder has the dim. adapter and 4 adapter layers, and the decoder follows MTee architecture (6 layers). *NLLB arch.* - encoder has no adapter, decoder follows NLLB-1B architecture (same vocabulary, embed. dim and ffn. dim. as encoder), the number in parenthesis denotes the number of decoder layers, embeddings are shared with encoder and frozen.

| Decoder | Params. | | Time | chrF++ |
|---|---|---|---|---|
| | trained | total | (hrs) | avg |
| MTee arch. | 51M | 817M | 4.3 | 42.8 |
| NLLB-1B arch. (6) | 151M | 917M | 7.4 | 42.5 |
| NLLB-1B arch. (2) | 50M | 816M | 5.8 | 38.2 |

### 5.5.3 The Effect of the Pre-trained Decoder

Given that the use of a pre-trained decoder yielded results comparable to a randomly initialized decoder, this thesis also looks into the speed of model convergence and the effect the amount of training data has on the results depending on the initialization.

To compare the convergence of models with different pre-trained decoder initialization, we train a model with a randomly initialized decoder, a decoder initialized from MTee, and also a variant of the latter with the decoder frozen. The encoder is initialized from NLLB and frozen. As illustrated in Figure 3, our findings reveal that the adapter-only training approach with a pre-trained encoder and decoder had surprisingly slow convergence for the first 2500 updates, even falling behind the randomly initialized decoder and faster convergence after that. However, when the decoder is also trained, we observed a faster convergence rate than with an uninitialized decoder.

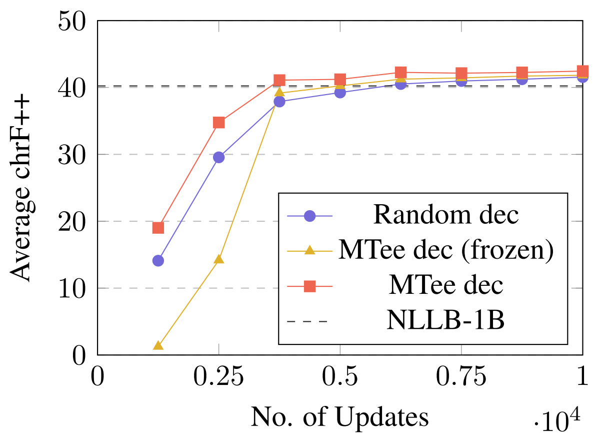

Figure 3. Average test chrF++ score for NLLB+MTee models for first 10,000 training updates (evaluated every 1250 updates). Decoder and adapter (dimensional and layers) are trained, with rest of the encoder frozen, except for MTee dec (frozen), where the decoder is also frozen.

To compare the effect of the dataset size for various initialization of decoders, we train a model with a randomly initialized decoder and a frozen MTee decoder. As with the previous experiment, we use a frozen NLLB decoder. The models are trained with the full dataset (53M sentence pairs in total), 1M sentence pairs sampled for each direction (4M in total), and 500k sentence pairs sampled for each direction (2M in total). As depicted in Figure 4, the model with a pre-trained encoder and a pre-trained decoder (MTee dec (frozen)) was less impacted by the dataset size than the model with only the encoder having been pre-trained. This is unsurprising as the MTee decoder has already been trained on a significant amount of data and thus likely requires less training.

### 5.5.4 The Effect of Adapter Module Structure and the Number of Languages

We explore the effect the number of languages and adapter module structure has on the results. Experiments were done with Europarl as the training set and the models were trained for 20 epochs on 1 GPU.

We can observe that using a dimension adapter without added layers yielded worse results than adding layers (experiments 1–6 in Table 13). We also experiment with an MLP (multi-layer perceptron) dimension adapter, which is composed in a similar way

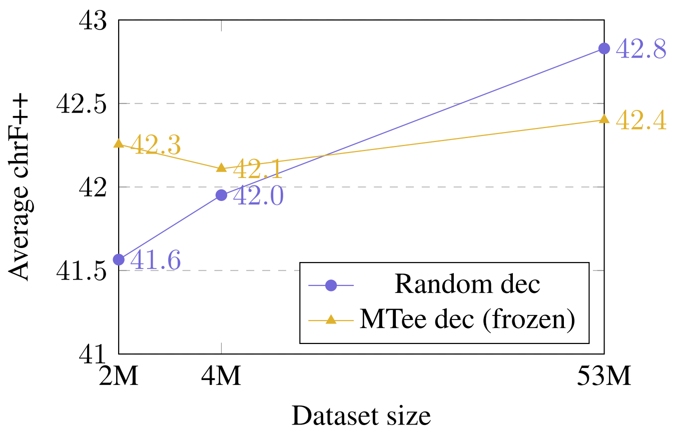

Figure 4. Average test chrF++ score for NLLB+MTee models for three dataset sizes: 500k sentence pairs per direction (2M in total), 1M per direction (4M in total) and the whole dataset (53M in total) trained for 100k updates. For MTee Dec model, only the dimensional adapter and adapter layers are trained, while the pre-trained decoder and encoder remain frozen.

to transformer feed-forward submodule – linear layer, activation (in our case ReLU), followed by another linear layer – in our case, the first linear layer changes the dimensionality. It can be seen that the experiment with MLP dimension adapter transformation only worked when no adapter layers were present. With adapter layers, training was unstable, and the results were lower, with large variances and confidence intervals. Therefore, it was better to use a linear dimension adapter.

We also investigated the impact of the dimension adapter position, but we did not observe any significant benefit from placing it before or between adapter layers (experiments 6 vs 7–8 in Table 13).

When considering the number of languages, we found that using four languages resulted in a slightly better score than using two (experiments 8-10 in Table 13). However, using six languages yielded no visible gain with the current dataset and configuration.

Regarding the number of adapter layers, we found that using four adapter layers resulted in the best score, although the improvement was likely not significant (experiments 9 and 11–13 in Table 13). It is worth noting that the results of determining how many layers to use might differ with the amount of data. When more capacity is required, a different number of layers may be necessary to achieve optimal results.

## 5.6 Using a Pre-trained Language Model

We additionally demonstrate the mix-and-match method with pre-trained language models by combining a pre-trained encoder with NMT decoder. We combine XLM-

Table 13. Many-to-Estonian translation chfF++ scores of ablation models trained on Europarl evaluated on FLORES-200 *devtest*. DA - dimension adapter, AL - adapter layer, DA + 2AL means dimension adapter followed by 2 adapter layers.

| ID | Model | | | chrF++ ↑ | |
|---|---|---|---|---|---|
| | | | | all | EU |
| | NLLB 600M baseline | | | 36.6 | 43.7 |
| | NLLB-600M + MTee | | | | |
| | adapter config | DA type | src langs | | |
| 1 | DA | MLP | en, de | 35.7 ± 0.2 | 43.2 ± 0.2 |
| 2 | DA | linear | en, de | 34.6 ± 0.3 | 42.2 ± 0.1 |
| 3 | DA + AL | MLP | en, de | 35.7 ± 2.3 | 43.3 ± 2.5 |
| 4 | DA + AL | linear | en, de | 38.2 ± 0.3 | 46.1 ± 0.3 |
| 5 | DA + 2 AL | MLP | en, de | 38.0 ± 1.9 | 45.7 ± 2.2 |
| 6 | DA + 2 AL | linear | en, de | 38.7 ± 0.3 | 46.5 ± 0.3 |
| 7 | 2 AL + DA | linear | en, de | 38.3 ± 0.9 | 46.2 ± 0.4 |
| 8 | AL + DA + AL | linear | en, de | 38.5 ± 0.2 | 46.4 ± 0.2 |
| 9 | DA + 2 AL | linear | en, de, fr, pl | 38.9 ± 0.1 | 46.7 ± 0.1 |
| 10 | DA + 2 AL | linear | en, de, fr, pl, lv, fi | 38.9 ± 0.1 | 46.7 ± 0.2 |
| 11 | DA + 3 AL | linear | en, de, fr, pl | 39.0 ± 0.1 | 46.8 ± 0.2 |
| 12 | DA + 4 AL | linear | en, de, fr, pl | 39.1 ± 0.1 | 46.8 ± 0.2 |
| 13 | DA + 5 AL | linear | en, de, fr, pl | 39.0 ± 0.2 | 46.7 ± 0.3 |

Roberta (Conneau et al., 2019) with MTee decoder. XLM-Roberta (XLM-R) supports fewer languages than NLLB, so we only evaluate the performance using European Union languages (except Maltese).

XLM-R + MTee is trained using two-stage training: First, training the adapter, and then the rest of the model. Here we don't keep the encoder frozen in the second stage, since it has not been trained for MT and is not as strong as the NLLB encoder for this task.

XLM-R + MTee achieves a significantly better score for 20 language pairs (68.2% of the directions) compared to the NLLB-1B baseline (see Table 14). However, on average, the score for XLM-R + MTee is worse than the NLLB-1B baseline for the EU languages. This could be potentially improved by adding more training data for the underperforming languages since right now our dataset has 4 source languages. Overall, these results are relatively good, considering the base model has not been trained for MT and has seen limited data and languages.

Table 14. Many-to-Estonian chrF++ and WRS (Win Ratio with Significance) scores for EU languages (except for Maltese). WRS - what percentage of directions have a better chrF++ score compared to the NLLB-1B baseline with $p = 0.01$ significance based on Paired Bootstrap Resampling t-test.

| Model | EU directions | |
|---|---|---|
| | WRS | avg chrF++ |
| NLLB-1B | - | 46.6 |
| NLLB+MTee, 2-stage | 100.0% | 50.8 |
| XLM-R-large+MTee, 2-stage | 68.2% | 46.0 |

We leave investigating the usage of a pre-trained decoder language model with our method for future work. Sun, Wang, and Li, 2021 has already investigated combining a BERT-like model (encoder side) and a GPT-like model (decoder side), and we see no obstacles in using their method of initializing part of the decoder from a GPT-like model.

## 5.7 Qualitative Evaluation

We also randomly choose sentences from the FLORES-200 test set to evaluate qualitatively. A brief look at some of the sentences reveals that all of the analyzed models offer relatively high translation quality, and it is difficult to make conclusions about the ranking of the models based on these translations without systematic human evaluation.

We give an example translation to show the relative translation quality of the systems and potential errors.

A translation from English to Estonian is given in Table 15. Firstly, it can be seen that all of the systems translate the source in a more direct and less creative way compared to the reference. Most of the translations are correct and very similar, however, the three that are clearly incorrect are MTee, NLLB fine-tune and NLLB-1B. MTee and NLLB fine-tune translated *pacifism* and *isolation* correctly, however, the meaning of the sentence is not correct due to grammatical errors. The NLLB-1B translates *pacifism* incorrectly and makes a grammatical mistake when translating *isolation*.

Table 15. Qualitative evaluation results.

| | |
|---|---|
| Source | During the 1920s, the prevailing attitudes of most citizens and nations was that of pacifism and isolation. |
| Reference | 1920. aastatel oli suur osa kodanikest ja rahvastest patsifistlikult ja isolatsionistlikult meelestatud. |
| NLLB-MoE | 1920. aastatel oli enamiku kodanike ja rahvaste valitsev hoiak patsifism ja isolatsioon. |
| NLLB-1B | 1920ndatel oli enamiku kodanike ja riikide valitsev suhtumine vaenlusele ja isoleerumisele. |
| MTee | 1920. aastatel valitses enamiku kodanike ja rahvaste suhtumine patsifismi ja isolatsiooni. |
| NLLB fine-tune | 1920. aastatel valitses enamiku kodanike ja rahvaste suhtumine patsifismisse ja isolatsiooni. |
| NLLB fine-tune decoder | 1920. aastatel oli enamiku kodanike ja rahvaste valitsev hoiak patsifism ja isolatsioon. |
| NLLB+MTee | 1920. aastatel oli enamiku kodanike ja rahvaste valitsevaks hoiakuks patsifism ja isolatsioon. |
| NLLB+MTee 2-stage | 1920. aastatel oli enamiku kodanike ja rahvaste valitsev hoiak patsifism ja isolatsioon. |

Looking closely at other NLLB+MTee model translations also reveals that the model will not know how to translate some symbols, since they are not part of the MTee vocabulary, for example, *"ğ"* in *"Erdoğan"*. A solution would be to add for example further normalization for punctuation and transliteration of missing characters. For example, *"Erdoğan"* could be written *"Erdogan"* to avoid the model ignoring the symbol or translating it as an unknown token symbol. Another option would be to add the missing symbols to the vocabulary and embedding weights of the decoder model and train the decoder ensuring that the desired symbols are also present in the training data.

# 6 Discussion

## 6.1 Possible Applications

As shown with the Ukrainian example, the proposed approach allows us to create a high-performing system in a matter of hours with limited data. This could be particularly important in crisis situations, for example, a sudden influx of refugees.

This approach is efficient from an environmental standpoint as well since it allows to reuse pre-trained models and avoids expensive re-training. It also helps to reduce the long-time costs of the model inference, since in the case demonstrated in this thesis, the number of parameters was reduced.

Furthermore, the proposed method also allows for a simple distributed way of training a modular translation system as it allows to independently train decoders for target languages or language groups and then combine them into one modular system while the pre-trained parts or just the encoder remain unchanged. Existing modular systems can be extended and combined as well through training adapter layers to unify the representations of different models or training new decoders. Possibly any transformer-based translation models could be combined with adapter layers to form a modular system.

The results of this thesis also demonstrate that a high-quality decoder is crucial in NMT. We have shown that even by training only the decoder and possibly a very limited amount of added encoder layers, it is possible to improve the translation quality, compared to a massively multilingual model.

## 6.2 Future Works

This work did not focus on training a one-to-many system and the few experiments that were carried out did not yield successful results. It should be investigated if it is possible to train such models with more available data and if there is any benefit from this kind of method from a cost and translation quality standpoint. Furthermore, it should be investigated if independently trained many-to-one decoder and one-to-many encoder that share the frozen pre-trained encoder/decoder respectively are compatible with each other. It is also an open question, how this method could be used to add translation from languages unsupported by the multilingual encoder.

This thesis conducted smaller proof-of-concept experiments, however, it is likely that they do not represent the ceiling of the performance with this method. It should be applied in a scenario with a larger dataset and longer training times to see the true potential of this approach. Furthermore, comparisons need to be made to the single, modular, and universal models in terms of training efficiency, translation quality, and inference times in a scenario with more resources.

The training efficiency of the proposed method also needs further comparisons to other parameter-efficient fine-tuning methods. Furthermore, parameter-efficient training methods could also be incorporated into the current approach: for example, instead of adapter layers, adapter submodules could be used before the optional dimension adapter. It also remains an open question of how will this approach perform on a domain different from the pre-trained encoder domain when the encoder is frozen during training. The parameter-efficient training of the encoder could also be a possible solution to this issue.

## 6.3 Limitations

One potential limiting factor of the proposed approach is the evaluation process. To ensure accurate and fair evaluation of the models, it is necessary to possess knowledge of the data on which the model was trained to avoid issues with leaky test data. The evaluation of our results relied primarily on automatic metrics, and we mainly utilized the FLORES-200 *devtest* due to the limited availability of test sets for Estonian and non-English languages. Additionally, we were unable to confirm that other available test sets were not part of the original models' training data, so we could not use them for a fair evaluation.

Moreover, the applicability of the mix-and-match method is dependent on the availability of pre-trained models in the target language. For instance, while Estonian models were readily available, other languages may not have such models, rendering the proposed method inapplicable. However, as an alternative, we proposed training the decoder from scratch and demonstrated its competitive performance.

It should also be noted that the translation quality results for Estonian cannot be generalized to all other languages. For example, English already exhibits high translation quality in most multilingual pre-trained NMT models, hence our method may not significantly improve performance as it would for Estonian. However, this limitation does not detract from other positive aspects of our method, including reduced parameter count and efficient training.

# 7 Conclusion

This thesis proposed and successfully demonstrated a novel method of combining encoders and decoders of pre-trained NMT models that would otherwise be incompatible with each other. The proposed method allows viewing encoders and decoders of pre-trained NMT models as modules that can be combined. It was demonstrated that this method of creating NMT models significantly improved translation quality judged by chrF++ scores and also reduced the number of parameters by 40%. Due to the reduced amount of parameters, this target-language-specific model requires less resources in the long term than the large massively multilingual model. The proposed method displayed high translation quality without the original parameters of the pre-trained models being changed, however, there was a slight increase in chrF++ scores when the decoder was allowed to continue training. The thesis additionally explored training new smaller decoders for pre-trained NMT models, which outperformed the original model both in terms of translation speed and quality. Furthermore, the thesis successfully showed the ability of the proposed methods to work for the rapid development of NMT models with limited data. It was also demonstrated that the proposed method would work with using pre-trained language model encoders as well.

## References

Aharoni, Roee, Melvin Johnson, and Orhan Firat (2019). “Massively multilingual neural machine translation”. In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*. Vol. 1. DOI: 10.18653/v1/n19-1388.

Arivazhagan, Naveen et al. (July 2019). “Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges”. In: URL: https://arxiv.org/abs/1907.05019.

Bapna, Ankur and Orhan Firat (2019). “Simple, scalable adaptation for neural machine translation”. In: *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*. DOI: 10.18653/v1/d19-1165.

Beltagy, Iz, Matthew E. Peters, and Arman Cohan (Apr. 2020). “Longformer: The Long-Document Transformer”. In: DOI: 10.48550/arxiv.2004.05150. URL: https://arxiv.org/abs/2004.05150.

Bergmanis, Toms and Marcis Pinnis (2022). “From Zero to Production: Baltic-Ukrainian Machine Translation Systems to Aid Refugees”. In: *Baltic Journal of Modern Computing* 10.3, pp. 271–282. URL: https://doi.org/10.22364/bjmc.2022.10.3.01.

Chen, Guanhua et al. (Nov. 2021). “Zero-shot Cross-lingual Transfer of Neural Machine Translation with Multilingual Pretrained Encoders”. In: *Proceedings of EMNLP*, pp. 15–26.

Chen, Guanhua et al. (2022). “Towards Making the Most of Multilingual Pretraining for Zero-Shot Neural Machine Translation”. In: *Proceedings of ACL*.

Chen, Peng-Jen et al. (2020). “Facebook AI’s WMT20 News Translation Task Submission”. In: *Proc. of WMT*.

Chi, Zewen et al. (June 2021). “InfoXLM: An Information-Theoretic Framework for Cross-Lingual Language Model Pre-Training”. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, pp. 3576–3588. DOI: 10.18653/v1/2021.naacl-main.280. URL: https://aclanthology.org/2021.naacl-main.280.

Conneau, Alexis et al. (2019). “Unsupervised Cross-lingual Representation Learning at Scale”. In: *arXiv preprint arXiv:1911.02116*.

Dabre, Raj, Chenhui Chu, and Anoop Kunchukuttan (2020). “A Survey of Multilingual Neural Machine Translation”. In: *ACM Computing Surveys* 53.5. ISSN: 15577341. DOI: 10.1145/3406095.

Dai, Zihang et al. (2020). “Transformer-XL: Attentive language models beyond a fixed-length context”. In: *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*. DOI: 10.18653/v1/p19-1285.

Devlin, Jacob et al. (2019). “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*. Vol. 1.

Dosovitskiy, Alexey et al. (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *International Conference on Learning Representations*. URL: `https://openreview.net/forum?id=YicbFdNTTy`.

Escolano, Carlos, Marta R. Costa-jussà, and José A. R. Fonollosa (2019). “From Bilingual to Multilingual Neural Machine Translation by Incremental Training”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 236–242. DOI: `10.18653/v1/P19-2033`.

Escolano, Carlos et al. (2021). “Multilingual Machine Translation: Closing the Gap between Shared and Language-specific Encoder-Decoders”. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 944–948. DOI: `10.18653/v1/2021.eacl-main.80`.

Fan, Angela et al. (Oct. 2020). “Beyond English-Centric Multilingual Machine Translation”. In: URL: `https://arxiv.org/abs/2010.11125`.

Fedus, William, Barret Zoph, and Noam Shazeer (2021). “Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity”. In: *J. Mach. Learn. Res* 23, pp. 1–40.

Freitag, Markus et al. (Dec. 2022). “Results of WMT22 Metrics Shared Task: Stop Using BLEU – Neural Metrics Are Better and More Robust”. In: *Proceedings of the Seventh Conference on Machine Translation (WMT)*. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, pp. 46–68. URL: `https://aclanthology.org/2022.wmt-1.2`.

Gage, P (1994). “A new algorithm for data compression”. In: *The C Users Journal archive* 12, pp. 23–38.

Gong, Hongyu, Xian Li, and Dmitriy Genzel (Apr. 2021). “Adaptive Sparse Transformer for Multilingual Translation”. In: URL: `http://arxiv.org/abs/2104.07358`.

Habash, Nizar and Jun Hu (2009). “Improving Arabic-Chinese Statistical Machine Translation using English as Pivot Language”. In: *EACL 2009 - 4th Workshop on Statistical Machine Translation, Proceedings of theWorkshop*. DOI: `10.3115/1626431.1626467`.

Johnson, Melvin et al. (2017). “Google’s Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation”. In: *Transactions of the Association for Computational Linguistics* 5. DOI: `10.1162/tacl{\_}a{\_}00065`.

Kingma, Diederik P. and Jimmy Lei Ba (2015). "Adam: A method for stochastic optimization". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Kocmi, Tom and Christian Federmann (Feb. 2023). "Large Language Models Are State-of-the-Art Evaluators of Translation Quality". In: DOI: 10.48550/arxiv.2302.14520. URL: https://arxiv.org/abs/2302.14520.

Koehn, Philipp (July 2004). "Statistical Significance Tests for Machine Translation Evaluation". In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, pp. 388–395. URL: https://aclanthology.org/W04-3250.

Kudo, Taku and John Richardson (2018). "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing". In: *EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings*. DOI: 10.18653/v1/d18-2012.

Lepikhin, Dmitry et al. (June 2020). "GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding". In: URL: http://arxiv.org/abs/2006.16668.

Liao, Junwei et al. (July 2021). "Improving Zero-shot Neural Machine Translation on Language-specific Encoders- Decoders". In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8. ISBN: 978-1-6654-3900-8. DOI: 10.1109/IJCNN52387.2021.9534401. URL: http://arxiv.org/abs/2102.06578.

Liu, Yinhan et al. (July 2019). "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: URL: https://arxiv.org/abs/1907.11692.

Liu, Yinhan et al. (2020). "Multilingual Denoising Pre-training for Neural Machine Translation". In: *Transactions of the Association for Computational Linguistics* 8, pp. 726–742. DOI: 10.1162/tacl{_}a{_}00343. URL: https://aclanthology.org/2020.tacl-1.47.

Lyu, Sungwon et al. (Nov. 2020). "Revisiting Modularized Multilingual NMT to Meet Industrial Demands". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 5905–5918. DOI: 10.18653/v1/2020.emnlp-main.476. URL: https://www.aclweb.org/anthology/2020.emnlp-main.476.

Ma, Shuming et al. (2021). "DeltaLM: Encoder-Decoder Pre-training for Language Generation and Translation by Augmenting Pretrained Multilingual Encoders". In.

Ng, Nathan et al. (2019). "Facebook FAIR's WMT19 News Translation Task Submission". In: *Proc. of WMT*.

NLLB Team et al. (2022). "No Language Left Behind: Scaling Human-Centered Machine Translation". In.

Ott, Myle et al. (2019). "Fairseq: A fast, extensible toolkit for sequence modeling". In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Associa-*

*tion for Computational Linguistics: Human Language Technologies - Proceedings of the Demonstrations Session*.

Papineni, Kishore et al. (2001). “BLEU: a Method for Automatic Evaluation of Machine Translation”. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL ’02*. Morristown, NJ, USA: Association for Computational Linguistics, p. 311. DOI: 10.3115/1073083.1073135.

Philip, Jerin et al. (Nov. 2020). “Monolingual Adapters for Zero-Shot Neural Machine Translation”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 4465–4470. DOI: 10.18653/v1/2020.emnlp-main.361. URL: https://aclanthology.org/2020.emnlp-main.361.

Popović, Maja (Sept. 2015). “chrF: character n-gram F-score for automatic MT evaluation”. In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Lisbon, Portugal: Association for Computational Linguistics, pp. 392–395. DOI: 10.18653/v1/W15-3049. URL: https://aclanthology.org/W15-3049.

— (Aug. 2016). “chrF deconstructed: beta parameters and n-gram weights”. In: *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*. Berlin, Germany: Association for Computational Linguistics, pp. 499–504. DOI: 10.18653/v1/W16-2341. URL: https://aclanthology.org/W16-2341.

— (2017). “CHRF++: Words helping character n-grams”. In: *WMT 2017 - 2nd Conference on Machine Translation, Proceedings*. DOI: 10.18653/v1/w17-4770.

Post, Matt (2018). “A Call for Clarity in Reporting BLEU Scores”. In: *WMT 2018 - 3rd Conference on Machine Translation, Proceedings of the Conference*. Vol. 1. DOI: 10.18653/v1/w18-6319.

Radford, Alec et al. (2019). “Language Models are Unsupervised Multitask Learners”. In.

Rei, Ricardo et al. (2020). “COMET: A neural framework for MT evaluation”. In: *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. DOI: 10.18653/v1/2020.emnlp-main.213.

Rei, Ricardo et al. (Dec. 2022). “COMET-22: Unbabel-IST 2022 Submission for the Metrics Shared Task”. In: *Proceedings of the Seventh Conference on Machine Translation (WMT)*. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, pp. 578–585. URL: https://aclanthology.org/2022.wmt-1.52.

Rothe, Sascha, Shashi Narayan, and Aliaksei Severyn (2020). “Leveraging Pre-trained Checkpoints for Sequence Generation Tasks”. In: *Transactions of the Association for Computational Linguistics* 8, pp. 264–280. DOI: 10.1162/tacl{_}a{_}00313. URL: https://aclanthology.org/2020.tacl-1.18.

Schwenk, Holger et al. (Nov. 2019). “CCMatrix: Mining Billions of High-Quality Parallel Sentences on the WEB”. In: URL: http://arxiv.org/abs/1911.04944.

Sellam, Thibault, Dipanjan Das, and Ankur Parikh (2020). “BLEURT: Learning Robust Metrics for Text Generation”. In: DOI: 10.18653/v1/2020.acl-main.704.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2016). “Neural machine translation of rare words with subword units”. In: *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*. Vol. 3. DOI: 10.18653/v1/p16-1162.

Shazeer, Noam et al. (Jan. 2017). “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer”. In: URL: http://arxiv.org/abs/1701.06538.

Song, Kaitao et al. (2019). “MASS: Masked Sequence to Sequence Pre-training for Language Generation”. In: *International Conference on Machine Learning*, pp. 5926–5936.

Sun, Zewei, Mingxuan Wang, and Lei Li (2021). “Multilingual Translation via Grafting Pre-trained Language Models”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: Findings*.

Tan, Xu et al. (2019). “Multilingual Neural Machine Translation with Knowledge Distillation”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=S1gUsoR9YX.

Tiedemann, Jörg and Santhosh Thottingal (2020). “OPUS-MT — Building open translation services for the World”. In: *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*. Lisbon, Portugal.

Tiedemann, Jörg (2012). “Parallel data, tools and interfaces in OPUS”. In: *Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC 2012*.

Tran, Chau et al. (2021). “Facebook AI’s WMT21 News Translation Task Submission”. In: *Proc. of WMT*.

Tättar, Andre et al. (2022). “Open and Competitive Multilingual Neural Machine Translation in Production. In: Proceedings of Baltic HLT 2022”. In: *Baltic Journal of Modern Computing* 10.3, 422434. URL: https://doi.org/10.22364/bjmc.2022.10.3.15.

University of Tartu (2018). *UT Rocket*. DOI: 10.23673/PH6N-0144. URL: https://share.neic.no/#/marketplace-public-offering/c8107e145e0d41f7a016b72825072287/.

Vaswani, Ashish et al. (2017). “Attention is all you need”. In: *Advances in Neural Information Processing Systems*. Vol. 2017-December.

Yang, Jian et al. (Nov. 2021). “Multilingual Machine Translation Systems from Microsoft for WMT21 Shared Task”. In: *Proceedings of the Sixth Conference on Machine Translation*. Online: Association for Computational Linguistics, pp. 446–455. URL: https://aclanthology.org/2021.wmt-1.54.

Zhang, Biao et al. (2020). “Improving Massively Multilingual Neural Machine Translation and Zero-Shot Translation”. In: DOI: 10.18653/v1/2020.acl-main.148.

Zhang, Biao et al. (2021). “Share or Not? Learning to Schedule Language-Specific Capacity for Multilingual Translation”. In: *International Conference on Learning Representations*. URL: `https://openreview.net/forum?id=Wj4ODo0uyCF`.

Zhu, Jinhua et al. (2020). “Incorporating BERT into Neural Machine Translation”. In: *International Conference on Learning Representations*. URL: `https://openreview.net/forum?id=Hyl7ygStwB`.

# Appendix

## I. Full Results

Table 16. Full many-to-Estonian translation chrF++ scores for NLLB+MTee mix-and-match model with 2-stage training and the baseline NLLB-1B-distilled model. NLLB+MTee 2-stage scores are averages of 5 seeds with $p = 0.01$ confidence interval.

| src lang | NLLB-1B | NLLB+MTee 2 stage | src lang | NLLB-1B | NLLB+MTee 2 stage |
|---|---|---|---|---|---|
| ace_Arab | 26.4 | 26.5 ± 0.4 | ace_Latn | 35.1 | 37.4 ± 0.8 |
| acm_Arab | 44.8 | 47.4 ± 0.1 | acq_Arab | 45.7 | 48.2 ± 0.4 |
| aeb_Arab | 42.7 | 44.9 ± 0.2 | afr_Latn | 48.5 | 52.4 ± 0.2 |
| ajp_Arab | 46.0 | 48.9 ± 0.3 | aka_Latn | 33.5 | 34.6 ± 1.3 |
| amh_Ethi | 41.6 | 44.7 ± 0.1 | apc_Arab | 45.1 | 47.7 ± 0.3 |
| arb_Arab | 45.8 | 49.1 ± 0.3 | ars_Arab | 45.8 | 48.5 ± 0.2 |
| ary_Arab | 40.2 | 42.8 ± 0.3 | arz_Arab | 43.3 | 45.9 ± 0.2 |
| asm_Beng | 40.4 | 42.8 ± 0.3 | ast_Latn | 44.4 | 48.4 ± 0.2 |
| awa_Deva | 44.7 | 47.0 ± 0.3 | ayr_Latn | 25.9 | 26.7 ± 0.8 |
| azb_Arab | 33.9 | 34.2 ± 1.6 | azj_Latn | 41.1 | 43.9 ± 0.2 |
| bak_Cyrl | 44.0 | 46.6 ± 0.5 | bam_Latn | 30.0 | 30.0 ± 1.2 |
| ban_Latn | 41.1 | 44.0 ± 0.5 | bel_Cyrl | 41.8 | 44.4 ± 0.2 |
| bem_Latn | 36.1 | 39.5 ± 0.3 | ben_Beng | 43.1 | 46.1 ± 0.4 |
| bho_Deva | 42.1 | 44.2 ± 0.3 | bjn_Arab | 28.6 | 29.7 ± 0.4 |
| bjn_Latn | 41.6 | 44.6 ± 0.4 | bod_Tibt | 29.2 | 30.8 ± 0.4 |
| bos_Latn | 47.9 | 52.8 ± 0.4 | bug_Latn | 33.5 | 34.8 ± 0.8 |
| bul_Cyrl | 47.8 | 51.7 ± 0.1 | cat_Latn | 48.1 | 52.0 ± 0.2 |
| ceb_Latn | 44.1 | 47.9 ± 0.3 | ces_Latn | 47.3 | 52.0 ± 0.4 |
| cjk_Latn | 24.7 | 25.6 ± 0.4 | ckb_Arab | 40.9 | 43.9 ± 0.1 |
| crh_Latn | 45.0 | 47.5 ± 0.3 | cym_Latn | 47.2 | 51.1 ± 0.1 |
| dan_Latn | 48.6 | 53.4 ± 0.3 | deu_Latn | 48.5 | 52.8 ± 0.2 |
| dik_Latn | 23.4 | 23.6 ± 1.0 | dyu_Latn | 22.3 | 22.6 ± 0.8 |
| dzo_Tibt | 31.3 | 32.2 ± 0.6 | ell_Grek | 45.0 | 48.5 ± 0.4 |
| eng_Latn | 52.6 | 57.3 ± 0.3 | epo_Latn | 48.9 | 52.6 ± 0.3 |
| eus_Latn | 43.6 | 47.6 ± 0.3 | ewe_Latn | 30.8 | 32.5 ± 0.6 |
| fao_Latn | 41.3 | 42.9 ± 1.4 | pes_Arab | 45.0 | 48.4 ± 0.3 |
| fij_Latn | 32.9 | 35.1 ± 0.5 | fin_Latn | 46.6 | 50.9 ± 0.2 |
| fon_Latn | 25.9 | 24.6 ± 1.9 | fra_Latn | 47.3 | 51.8 ± 0.2 |
| fur_Latn | 46.6 | 50.5 ± 0.3 | fuv_Latn | 25.3 | 25.7 ± 0.9 |

*The table continues on the next page.*

*The table is continued from the previous page.*

| src lang | NLLB-1B | NLLB+MTee 2 stage | src lang | NLLB-1B | NLLB+MTee 2 stage |
|---|---|---|---|---|---|
| gla_Latn | 39.9 | 42.7 ± 0.3 | gle_Latn | 43.2 | 47.2 ± 0.3 |
| glg_Latn | 47.5 | 51.2 ± 0.3 | grn_Latn | 36.1 | 38.1 ± 0.5 |
| guj_Gujr | 45.0 | 47.8 ± 0.2 | hat_Latn | 43.3 | 47.3 ± 0.3 |
| hau_Latn | 39.4 | 42.4 ± 0.3 | heb_Hebr | 46.4 | 50.7 ± 0.2 |
| hin_Deva | 45.3 | 48.2 ± 0.3 | hne_Deva | 45.6 | 49.1 ± 0.3 |
| hrv_Latn | 46.5 | 50.9 ± 0.2 | hun_Latn | 45.6 | 49.8 ± 0.3 |
| hye_Armn | 46.2 | 50.3 ± 0.3 | ibo_Latn | 37.3 | 40.1 ± 0.1 |
| ilo_Latn | 42.2 | 45.7 ± 0.3 | ind_Latn | 45.8 | 49.9 ± 0.1 |
| isl_Latn | 41.8 | 45.7 ± 0.4 | ita_Latn | 45.4 | 48.9 ± 0.2 |
| jav_Latn | 42.4 | 45.8 ± 0.2 | jpn_Jpan | 41.1 | 43.2 ± 0.3 |
| kab_Latn | 34.4 | 35.2 ± 0.3 | kac_Latn | 30.5 | 31.0 ± 1.6 |
| kam_Latn | 26.6 | 29.0 ± 0.4 | kan_Knda | 42.2 | 45.3 ± 0.3 |
| kas_Arab | 41.2 | 43.4 ± 0.2 | kas_Deva | 34.8 | 36.2 ± 0.2 |
| kat_Geor | 42.8 | 46.2 ± 0.2 | knc_Arab | 14.5 | 12.6 ± 1.6 |
| knc_Latn | 28.1 | 29.1 ± 0.2 | kaz_Cyrl | 44.7 | 47.9 ± 0.5 |
| kbp_Latn | 28.7 | 30.6 ± 0.6 | kea_Latn | 45.2 | 49.5 ± 0.2 |
| khm_Khmr | 41.3 | 43.6 ± 0.1 | kik_Latn | 33.1 | 37.2 ± 0.5 |
| kin_Latn | 39.4 | 42.7 ± 0.2 | kir_Cyrl | 39.5 | 42.8 ± 0.4 |
| kmb_Latn | 26.2 | 28.5 ± 0.1 | kon_Latn | 32.9 | 36.4 ± 0.3 |
| kor_Hang | 41.7 | 44.1 ± 0.2 | kmr_Latn | 37.7 | 40.0 ± 0.3 |
| lao_Laoo | 41.9 | 44.6 ± 0.3 | lvs_Latn | 45.8 | 50.6 ± 0.2 |
| lij_Latn | 45.7 | 49.9 ± 0.3 | lim_Latn | 44.1 | 47.0 ± 0.2 |
| lin_Latn | 37.2 | 40.0 ± 0.2 | lit_Latn | 45.0 | 49.1 ± 0.2 |
| lmo_Latn | 44.1 | 46.6 ± 0.3 | ltg_Latn | 46.3 | 49.8 ± 0.3 |
| ltz_Latn | 47.5 | 51.8 ± 0.3 | lua_Latn | 30.8 | 33.1 ± 0.3 |
| lug_Latn | 33.0 | 35.1 ± 0.2 | luo_Latn | 33.6 | 37.4 ± 0.3 |
| lus_Latn | 31.6 | 32.9 ± 0.2 | mag_Deva | 46.0 | 49.2 ± 0.2 |
| mai_Deva | 45.0 | 48.2 ± 0.2 | mal_Mlym | 43.2 | 46.3 ± 0.2 |
| mar_Deva | 44.0 | 46.9 ± 0.3 | min_Latn | 42.2 | 44.1 ± 0.3 |
| mkd_Cyrl | 47.4 | 51.7 ± 0.2 | plt_Latn | 40.9 | 44.6 ± 0.3 |
| mlt_Latn | 48.8 | 53.2 ± 0.3 | mni_Beng | 36.9 | 38.8 ± 0.2 |
| khk_Cyrl | 39.7 | 42.5 ± 0.2 | mos_Latn | 25.8 | 25.9 ± 1.3 |
| mri_Latn | 36.9 | 39.3 ± 0.2 | zsm_Latn | 45.5 | 48.7 ± 0.4 |
| mya_Mymr | 39.9 | 42.7 ± 0.3 | nld_Latn | 43.9 | 47.3 ± 0.2 |
| nno_Latn | 46.7 | 49.4 ± 0.3 | nob_Latn | 44.9 | 47.5 ± 0.3 |
| npi_Deva | 44.8 | 47.9 ± 0.2 | nso_Latn | 41.0 | 43.6 ± 0.3 |

*The table continues on the next page.*

*The table is continued from the previous page.*

| src lang | NLLB-1B | NLLB+MTee 2 stage | src lang | NLLB-1B | NLLB+MTee 2 stage |
|---|---|---|---|---|---|
| nus_Latn | 29.4 | 30.2 ± 0.5 | nya_Latn | 36.8 | 40.1 ± 0.3 |
| oci_Latn | 49.1 | 53.5 ± 0.2 | gaz_Latn | 35.4 | 37.4 ± 0.2 |
| ory_Orya | 43.3 | 46.4 ± 0.4 | pag_Latn | 36.5 | 39.0 ± 0.2 |
| pan_Guru | 44.9 | 47.4 ± 0.1 | pap_Latn | 47.4 | 51.3 ± 0.4 |
| pol_Latn | 43.8 | 47.9 ± 0.4 | por_Latn | 48.5 | 52.4 ± 0.3 |
| prs_Arab | 45.3 | 47.5 ± 0.2 | pbt_Arab | 41.9 | 44.3 ± 0.1 |
| quy_Latn | 26.5 | 27.9 ± 0.6 | ron_Latn | 48.1 | 52.5 ± 0.3 |
| run_Latn | 37.1 | 39.8 ± 0.2 | rus_Cyrl | 46.6 | 50.4 ± 0.3 |
| sag_Latn | 27.2 | 28.2 ± 0.6 | san_Deva | 35.5 | 37.2 ± 0.5 |
| sat_Olck | 30.5 | 30.3 ± 0.5 | scn_Latn | 43.7 | 47.0 ± 0.1 |
| shn_Mymr | 36.0 | 37.8 ± 0.4 | sin_Sinh | 42.5 | 46.1 ± 0.1 |
| slk_Latn | 47.6 | 52.0 ± 0.2 | slv_Latn | 46.0 | 50.3 ± 0.3 |
| smo_Latn | 38.7 | 42.3 ± 0.2 | sna_Latn | 36.8 | 39.7 ± 0.2 |
| snd_Arab | 43.8 | 46.1 ± 0.3 | som_Latn | 38.4 | 40.2 ± 0.2 |
| sot_Latn | 42.4 | 46.0 ± 0.3 | spa_Latn | 44.5 | 47.6 ± 0.2 |
| als_Latn | 46.8 | 51.2 ± 0.2 | srd_Latn | 46.0 | 49.0 ± 0.1 |
| srp_Cyrl | 47.6 | 52.5 ± 0.2 | ssw_Latn | 35.9 | 40.2 ± 0.1 |
| sun_Latn | 43.4 | 46.1 ± 0.3 | swe_Latn | 48.3 | 52.7 ± 0.1 |
| swh_Latn | 43.9 | 47.9 ± 0.2 | szl_Latn | 47.2 | 51.6 ± 0.1 |
| tam_Taml | 42.2 | 45.1 ± 0.3 | tat_Cyrl | 43.5 | 47.4 ± 0.2 |
| tel_Telu | 43.5 | 46.9 ± 0.2 | tgk_Cyrl | 44.2 | 47.7 ± 0.2 |
| tgl_Latn | 45.8 | 49.1 ± 0.4 | tha_Thai | 41.4 | 44.7 ± 0.3 |
| tir_Ethi | 36.1 | 38.6 ± 0.3 | taq_Latn | 25.7 | 26.2 ± 0.7 |
| taq_Tfng | 20.9 | 18.9 ± 1.4 | tpi_Latn | 36.6 | 39.4 ± 0.4 |
| tsn_Latn | 37.5 | 40.8 ± 0.2 | tso_Latn | 38.8 | 42.0 ± 0.1 |
| tuk_Latn | 43.9 | 47.1 ± 0.3 | tum_Latn | 33.7 | 36.6 ± 0.3 |
| tur_Latn | 45.3 | 49.5 ± 0.1 | twi_Latn | 33.6 | 35.9 ± 0.3 |
| tzm_Tfng | 31.0 | 32.2 ± 0.3 | uig_Arab | 39.7 | 42.0 ± 0.2 |
| ukr_Cyrl | 47.0 | 51.0 ± 0.2 | umb_Latn | 25.7 | 27.3 ± 0.3 |
| urd_Arab | 43.5 | 46.5 ± 0.4 | uzn_Latn | 44.6 | 48.6 ± 0.2 |
| vec_Latn | 45.8 | 49.2 ± 0.3 | vie_Latn | 43.7 | 47.4 ± 0.2 |
| war_Latn | 44.6 | 49.2 ± 0.1 | wol_Latn | 30.4 | 30.9 ± 1.2 |
| xho_Latn | 40.5 | 43.8 ± 0.3 | ydd_Hebr | 44.6 | 47.1 ± 0.2 |
| yor_Latn | 32.6 | 34.2 ± 0.3 | yue_Hant | 39.8 | 44.1 ± 0.3 |
| zho_Hans | 40.2 | 44.6 ± 0.4 | zho_Hant | 38.3 | 43.4 ± 0.2 |
| zul_Latn | 41.7 | 45.4 ± 0.1 | | | |

## II. Language Codes

Table 17. The language codes used in this thesis. We either use a shorter 2-letter code or the code from NLLB (NLLB Team et al., 2022), which provides less ambiguity.

| language | script | code | |
|---|---|---|---|
| | | 2-letter | NLLB |
| Estonian | Latin | ET | est_Latn |
| English | Latin | EN | eng_Latn |
| German | Latin | DE | deu_Latn |
| Russian | Cyrillic | RU | rus_Cyrl |
| Ukrainian | Cyrillic | UK | ukr_Cyrl |
| Standard Latvian | Latin | LV | lvs_Latn |
| Lithuanian | Latin | LT | lit_Latn |
| Polish | Latin | PL | pol_Latn |
| French | Latin | FR | fra_Latn |
| Modern Standard Arabic | Arabic | AR | arb_Arab |
| Chinese | Han (Simplified) | ZH | zho_Hans |

## III. Licence