Karlsruhe Institute of Technology

AI4LT

Zero-shot and few-shot multilingual abstractive summarization

Master's Thesis of

Vladimir Solovyev

Artificial Intelligence for Language Technologies (AI4LT) Lab
Institut for Anthropomatics and Robotics (IAR)
KIT Department of Informatics

Reviewer:	Prof. Dr. Jan Niehues
Second reviewer:	Prof. Dr. Alexander Waibel
Advisor:	M.Sc. Danni Liu

01.06.2023 – 01.12.2023

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

I declare that I have developed and written the enclosed thesis completely by myself. I have submitted neither parts of nor the complete thesis as an examination elsewhere. I have not used any other than the aids that I have mentioned. I have marked all parts of the thesis that I have included from referenced literature, either in their original wording or paraphrasing their contents. This also applies to figures, sketches, images and similar depictions, as well as sources from the internet.

PLACE, DATE

. . . . . .

(Vladimir Solovyev)

Abstract

Fine-tuning pre-trained multilingual models, such as mBART, enables the achievement of promising results in multilingual abstractive text summarization. However, it can be challenging for language pairs with limited or no provided supervised data, which is often the case for low-resource languages.

This thesis investigates different methods to improve the quality of zero-shot multilingual abstractive text summarization using pre-trained multilingual models, e.g., mBART. Specifically, I propose fine-tuning only queries and keys of multi-head attention, resulting in better results and requiring less time for fine-tuning as it updates fewer parameters. Applying this method during fine-tuning the mBART model with only English data makes it possible to perform intralingual summaries in other languages, including low-resource languages. For example, for Gujarati, it improves the BERTScore-F1 metric from 59.2 to 73.2 compared to the model fine-tuned without this method.

Other approaches proposed in this thesis enable cross-lingual summarization in zero-shot cases by fine-tuning the mBART model using only monolingual data in multiple languages. Language-specific encoder output adapters adapt encoder representation tokens to the output language and force the output to be in the expected language. This approach performs well but is slightly less effective than another approach that uses an adversarial language classifier. The use of the adversarial language classifier makes the encoder output less language-specific. In my thesis, I also present an updated version of this approach that better encourages language-independent encoder output representation by using Kullback–Leibler divergence to a uniform class distribution as a loss function. Additionally, I combine this updated version with another approach that removes a residual connection from one encoder layer to decrease positional information from the encoder output. This combination of methods produces the best results. For example, in the Russian-English pair, the Rouge-1 value improves from 31.9 to 33.1. I also analyzed the encoder output representation, attempting to recognize the input language. This analysis confirms that new approaches lead to more language-independent encoder output representation results.

This thesis also determines the required amount of supervised data for few-shot experiments with which it is possible to achieve results comparable to experiments with complete supervised data in both intralingual and cross-lingual scenarios. Experiments showed that even a small amount of supervised data in the expected languages (10 samples) can significantly improve the results. Adding more data further improves the results. In most scenarios, fine-tuning with 1000 or 10000 samples leads to results close to the supervised solution.

Zusammenfassung

Die Fine-tuning von vortrainierten mehrsprachigen Modellen wie mBART ermöglicht versprechende Ergebnisse bei der mehrsprachigen abstrakten Textzusammenfassung. Allerdings kann dies bei Sprachpaaren mit begrenzten oder gar keinen überwachten Daten eine Herausforderung darstellen, was bei Low-Resource-Sprachen oft der Fall ist.

In dieser Arbeit werden verschiedene Methoden zur Verbesserung der Qualität mehrsprachiger abstrakter Textzusammenfassungen in Zero-Shot-Fällen mit vortrainierten mehrsprachigen Modellen, z.B. mBART, untersucht. Ich schlage vor, nur die Queries und Keys der Multi-Head-Attention zu fine-tunen, was zu besseren Ergebnissen führt und weniger Zeit für die Fine-tuning erfordert, da weniger Parameter aktualisiert werden. Die Anwendung dieser Methode während der Fine-tuning des mBART-Modells mit ausschließlich englischen Daten ermöglicht die Durchführung von intralingualen Zusammenfassungen in anderen Sprachen, einschließlich Low-Resource Sprachen. Für Gujarati beispielsweise verbessert sie die BERTScore-F1-Metrik von 59,2 auf 73,2 im Vergleich zu dem Modell, das ohne diese Methode fine-tuned wurde.

Andere Ansätze, die in dieser Abschlussarbeit vorgeschlagen werden, ermöglichen eine sprachübergreifende Zusammenfassung in Zero-Shot-Fällen, indem das mBART-Modell nur mit monolingualen Daten in mehreren Sprachen fine-getuned wird. Sprachspezifische Encoder-Output-Adapter passen die Token der Encoder-Repräsentation an die Ausgabesprache an und erzwingen die Ausgabe in der erwarteten Sprache. Dieser Ansatz erbringt gute Leistungen, ist aber etwas weniger effektiv als ein anderer Ansatz, der einen Adversarial-Language-Classifier verwendet. Durch den Einsatz des Adversarial-Language-Classifier ist die Encoder-Repräsentation weniger sprachspezifisch. In meiner Abschlussarbeit stelle ich auch eine aktualisierte Version dieses Ansatzes vor, die die Encoder-Repräsentation noch weniger sprachspezifisch macht und die Kullback-Leibler-Divergenz zu einer uniformen Klassendistribution als Verlustfunktion verwendet. Außerdem kombiniere ich diese aktualisierte Version mit einem anderen Ansatz, bei dem eine Residual-Verbindung einer Encoderschicht entfernt wird, um die Positionsinformationen aus der Encoder-Repräsentation zu verringern. Diese Kombination von Methoden führt zu den besten Ergebnissen. So verbessert sich beispielsweise der Rouge-1-Wert für das russisch-englische Paar von 31,9 auf 33,1. Ich habe auch die Darstellung der Encoder-Repräsentation analysiert und versucht, die Eingabesprache zu erkennen. Diese Analyse bestätigt, dass die neuen Ansätze zu ergebnissen führen, die eine sprachunabhängigere Encoder-Repräsentation zeigen.

In dieser Abschlussarbeit wird auch die erforderliche Menge an überwachten Daten für Few-Shot-Experimente bestimmt, mit denen es möglich ist, sowohl in intralingualen

als auch in sprachübergreifenden Szenarien vergleichbare Ergebnisse wie mit vollständig überwachten Daten zu erzielen. Die Experimente haben gezeigt, dass bereits eine kleine Menge überwachter Daten in den erwarteten Sprachen (10 Stichproben) die Ergebnisse deutlich verbessern kann. Durch Hinzufügen weiterer Daten werden die Ergebnisse weiter verbessert. In den meisten Szenarien führt die Fine-tuning mit 1000 oder 10000 Stichproben zu Ergebnissen, die der überwachten Lösung nahe kommen.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Text summarization helps analyze large volumes of data in a short time. It enables people to obtain relevant key information faster without reading entire texts. It can be applied in different use cases. For example, when reading research papers, we first read abstracts to understand what important information these papers contain. News summarization can provide readers with key facts from different articles. Summarizing customer reviews helps to receive faster feedback and understand if customers are satisfied. Text summarization can also be applied as a pre-processing step for other tasks, e.g., information retrieval, facilitating the search process.

Abstractive text summarization is a sequence-to-sequence task in natural language processing where longer texts are inputted, and shorter versions containing key information are generated. Unlike extractive methods, which only extract sentences or phrases from the original texts, abstractive summarization creates new texts that may include novel words, phrases, and sentences.

Multilingual abstractive summarization involves working with input texts and summaries in various languages, including multiple intralingual (input and output in the same language) input text-summary pairs and even multiple cross-lingual (input and output languages can differ) input text-summary pairs. Training models for performing multilingual abstractive summarization requires much more supervised data in different languages, making it more challenging. This is especially problematic for low-resource languages and different cross-lingual combinations of languages for which such data does not exist. It complicates the training of separate models for every language pair. Also, this approach exponentially increases the number of required models $O(n^2)$ (where n is a number of languages) and extends training time.

Another approach for solving the multilingual summarization task is to combine multiple neural network models. Firstly, it is necessary to train an intralingual summarization model that, for example, generates summaries in English from texts in English. Secondly, other translation models can be utilized for translation to and from English and other languages for data pre- and post-processing. In this case, only one summarization model needs to be trained, and it requires supervised data only in one language. However, this approach suffers from error propagation in each model, leading to overall poor performance.

An alternative solution that addresses these drawbacks involves using a single multilingual model capable of generating intra- and cross-lingual summaries in different

languages. This approach utilizes transfer learning between languages and eliminates the need for supervised data for every language pair. Pre-trained models can be used to enhance the quality of such a model and enable zero-shot summarization for low-resource languages that were not included in the training data. One such model is mBART presented by Yinhan Liu et al. (2020). The mBART model uses the Transformer architecture introduced by Vaswani et al. (2017), which represents a state-of-the-art solution for sequence-to-sequence tasks in natural language processing. The mBART model is pre-trained on a huge amount of intralingual data across 25 languages, including high-resource and low-resource. By fine-tuning pre-trained multilingual models such as mBART, it becomes possible to generate mono- and cross-lingual summaries in different languages.

The challenges faced in multilingual summarization are similar to those in machine translation. In machine translation, various methods have been investigated to achieve better results in zero-shot and few-shot cases (Arivazhagan et al. 2019; D. Liu, Niehues, et al. 2021; Philip et al. 2020). The objective of this thesis is to adapt these methods to multilingual summarization and evaluate their impact on the quality of zero-shot and few-shot summarizations. By conducting few-shot experiments on pre-trained models, the aim is to determine the amount of data required to achieve results comparable to experiments conducted with complete supervised data.

1.2 Problem statement and research questions

Abstractive summarization can be performed by deep neural networks using an encoder-decoder architecture, particularly the Transformer model proposed by Vaswani et al. (2017). However, training such models from scratch with a large amount of multilingual data can be very expensive and requires supervised data in numerous languages. To address this issue and improve both the quality of summarization and the speed of training, pre-trained Transformer models can be utilized. Specifically, for multilingual tasks like multilingual abstractive summarization, employing pre-trained models that have been trained on extensive multilingual data can be highly beneficial. These models are capable of encoding and generating high-quality texts in various languages, including both high-resource and low-resource languages.

The utilization of multiple languages during pretraining enables the identification of common features and facilitates transfer learning across languages, similar to how humans operate. Just as a person who speaks multiple languages can learn to create text summaries in one language and subsequently apply that knowledge to summarize texts in another language more efficiently, multilingual models can be fine-tuned using data from a single language and then used for summarization in unseen languages during the fine-tuning process. In this thesis, the performance of fine-tuning pre-trained multilingual models for the multilingual abstractive summarization task will be investigated, leading to the formulation of the following research questions:

Research question 1: How can we generate intralingual summaries of texts in low-resource languages when we only have intralingual data available in other languages and a pre-trained multilingual model?

Research question 2: How can we generate cross-lingual summaries of texts when we only have intralingual data available in the same languages or other languages, along with a pre-trained multilingual model? Various scenarios and cases need to be considered in this context:

- The languages are available as intralingual data during the fine-tuning process, but they are considered zero-shot for cross-lingual summarization.
- The output language is available during the fine-tuning process, but the input language is not.
- The input language is available during the fine-tuning process, but the output language is not.

Research question 3: What is the required amount of supervised data for few-shot experiments to achieve results that are comparable to experiments with complete supervised data in both intralingual and cross-lingual scenarios?

Research question 4: What methods can be applied to improve zero-shot results in both intralingual and cross-lingual scenarios?

1.3 Thesis outline

This thesis has the following structure. Chapter 2 provides background information. It begins with a description of the summarization task, followed by an explanation of how Transformer models work. Chapter 3 introduces various methods from the existing literature that can improve the results of multilingual experiments. Chapter 4 describes approaches that I propose in this thesis that could lead to further improvements. Chapter 5 provides an experimental setup that I use for experiments, for example, hyperparameters and metrics that I apply. Chapter 6 and Chapter 7 explain conducted experiments and discusses their results. Chapter 8 summarizes the results of my thesis, addresses the research questions and offers suggestions for further investigation.

2 Background

2.1 Sequence-to-sequence solution architecture

Recurrent neural networks (Rumelhart, Hinton, and Williams 1986), including gated recurrent neural networks (Chung et al. 2014) and long short-term memory (Hochreiter and Schmidhuber 1997), were considered state-of-the-art solutions for sequence-to-sequence problems according to Vaswani et al. (2017). However, their main drawback is the sequential nature of computations, which leads to longer training times. In 2017, a new model architecture called the Transformer was introduced by Vaswani et al. (2017). This innovative approach allows significantly more parallelization and has emerged as the new state-of-the-art solution for sequence-to-sequence problems, including abstractive summarization.

The Transformer model is an attention-based encoder-decoder architecture. The Transformer encoder takes an input text which should be summarized as a sequence of tokens x = $[x_1; \ldots; x_n]$ and maps it to a sequence of continuous representations z = $[z_1; \ldots; z_n]$. The encoder consists of a stack of N identical layers. Every layer includes two parts. The first is a multi-head self-attention mechanism, and the second is a fully connected feed-forward network. A residual connection is used around each part followed by a normalization layer. The output of each part is $LayerNorm(x + Part(x))$, where $Part(x)$ is the function implemented by each part.

The Transformer decoder also consists of a stack of N identical layers. In addition to the two parts in each encoder layer, the decoder contains a third part, which performs multi-head attention over the output of the encoder. The Transformer decoder takes a continuous representation z, along with all previously generated tokens, as input and generates the output summary $y = [y_1; \ldots; y_m]$ token by token (Vaswani et al. 2017). At each step, the model is auto-regressive and calculates conditional probabilities P($y_1; \ldots; y_m \mid x_1; \ldots; x_n$) (Yang Liu and Lapata 2019). The conditional probability of each token indicates the likelihood of that token given the specific input text and all previously generated tokens. Figure 1 provides its architecture.

The attention mechanism plays an important role in the Transformer architecture. It analyzes sequences by paying more or less attention to different positions of sequences depending on their importance.

The Transformer attention mechanism is referred to as multi-head attention. "Multi-head" means that it operates with h attentions or heads in parallel. As input, the multi-head attention receives three vectors at each position: query, key, and value. Firstly, these vectors

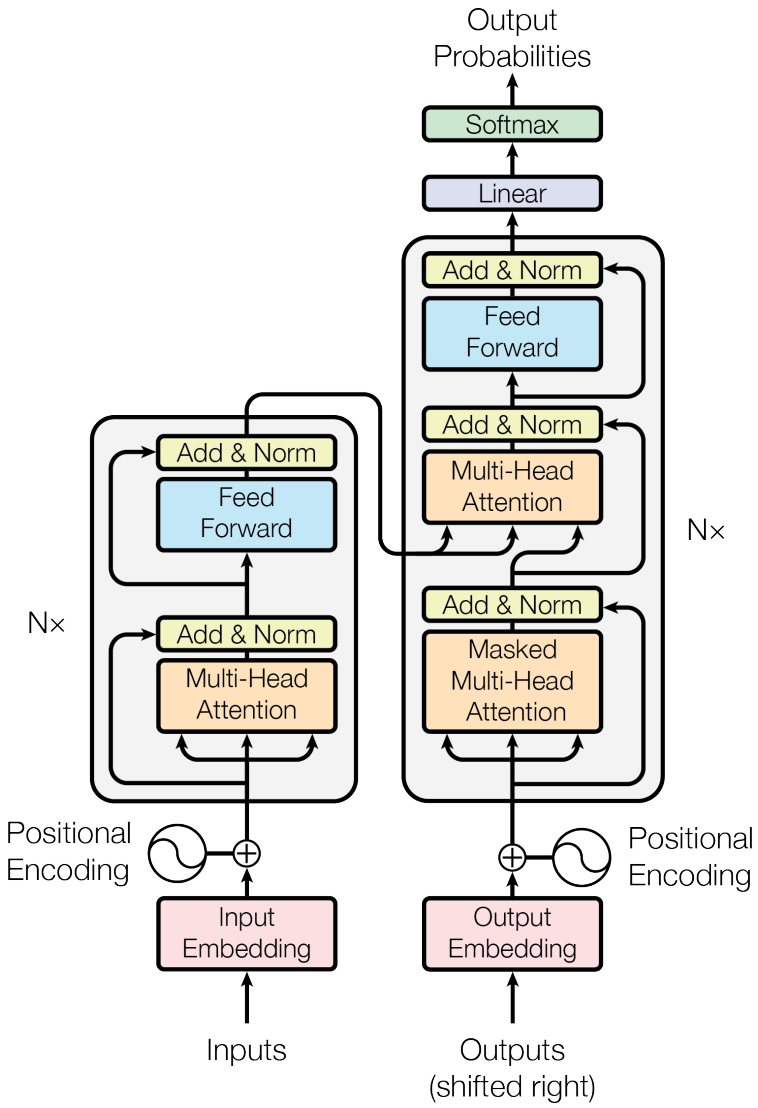

Figure 1: The Transformer - model architecture (Vaswani et al. 2017).

are linearly projected h times into dimensions d_q, d_k, and d_v using distinct learned linear projections. Secondly, a scaled dot-product attention is applied in parallel to the projected vectors, generating output values of dimension d_v for each head. Thirdly, the output values from h heads are concatenated into a single vector. Finally, this concatenated vector is linearly projected to obtain the output vector, which represents the final values of the multi-head attention. Figure 2 illustrates the structure of the multi-head attention.

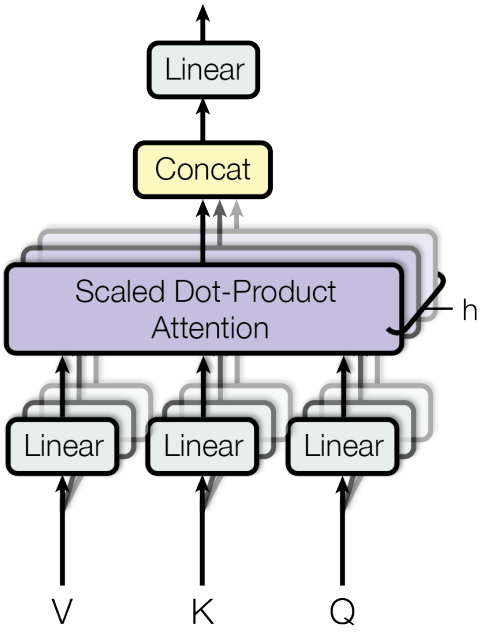

Figure 2: Multi-Head Attention (Vaswani et al. 2017).

$$
\begin{aligned}
\text{MultiHead}(Q,K,V) &= \text{Concat}(\text{head}_1,...,\text{head}_\text{h})W^O,\\
\text{where head}_\text{i} &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V),\\
W_i^Q &\in \mathbb{R}^{d_{model}\times d_k},\\
W_i^K &\in \mathbb{R}^{d_{model}\times d_k},\\
W_i^V &\in \mathbb{R}^{d_{model}\times d_v},\\
W^O &\in \mathbb{R}^{hd_v\times d_{model}}\\
\text{Attention}(Q,K,V) &= \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V
\end{aligned}
\tag{1}
$$

An advantage of multi-head attention is that it allows the model to learn information from different representation subspaces at different positions. With a single attention head, it is impossible because of the averaging of values.

A part of multi-head attention that performs attention for every head is called scaled dot-product attention. As input, it receives queries, keys, and values of dimensions d_q, d_k, and d_v respectively. Firstly, it calculates the dot products between the query and all keys. Secondly, it divides the results by $\sqrt{d_k}$. After that, a softmax function is applied, generating weights for the input values. Finally, it multiplies the input values by these weights and computes the sum. All of these computations are performed in parallel for input positions

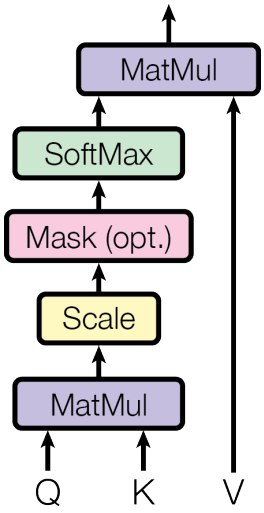

Figure 3: Scaled Dot-Product Attention (Vaswani et al. 2017).

using matrices that contain all input vectors, Q, K, and V. Figure 3 presents the structure of scaled dot-product attention.

In Transformer architecture attention is utilized in three places:

- Self-attention in encoder layers: In this case, input queries, keys, and values are identical and represent the output of the previous layer in the encoder. Each position in the encoder can use vectors from all other positions, both before and after itself.
- Self-attention in decoder layers: Similar to self-attention in encoder layers, but it cannot utilize vectors from positions that come after itself because a decoder is auto-regressive. This property is implemented by masking out all positions to the right of the current position before applying the softmax function in the scaled dot-product attention. The masking is achieved by setting values to -∞, resulting in zero probability after applying the softmax function.
- Encoder-attention in decoder layers: In this case, keys and values are derived from the output of the encoder, while only queries represent the output of the previous layer in the decoder. This enables every position in the decoder to obtain information from all positions in the input sequence.

2.2 Pre-training for sequence-to-sequence tasks

Training a Transformer model for solving an NLP task from scratch can be challenging, requiring substantial hardware resources and time. Additionally, it needs a significant amount of data to enable the model to acquire knowledge about various language patterns, structures, semantic understanding, and contextual representations of words. When working with multilingual models, even more data is required to learn to encode and generate texts in different languages. Supervised datasets used for many tasks may not be sufficient to understand languages and different relationships due to their potentially limited size.

One potential solution to this challenge is the use of pre-trained models, which are widely employed for NLP tasks nowadays. These models are typically trained on huge amounts of unsupervised data, making it easier to obtain data. After being trained for a certain period, pre-trained models can encode and generate texts well, although they are not yet ready for solving specific NLP tasks. To adapt these models for various NLP tasks such as question answering, machine translation, text summarization, etc., they need to be fine-tuned using task-specific supervised data.

An advantage of fine-tuning pre-trained models is that this approach requires less supervised data and training time to achieve acceptable results compared to training from scratch. This makes the training process faster and more resource-efficient. The effectiveness of this approach lies in transfer learning between tasks, as many tasks share similarities and involve similar subtasks. All sequence-to-sequence models, including pre-trained models, are trained to understand input texts and generate corresponding output texts. Hence, fine-tuning necessitates less data and fewer resources.

Different types of pre-trained models are based on complete Transformers architecture or only some components. Wang et al. (2022) describe these types:

- **Encoder-only models**. Such models utilize only an encoder from the Transformers architecture. These models are a good choice for an analysis of sequence representations. They can perform tasks at the sentence or token level such as natural language inference, named entity recognition, and question-answering. An example of such a model is BERT (Bidirectional Encoder Representations from Transformers), as proposed by Devlin et al. (2019).
- **Decoder-only models**. Such models utilize only a decoder from the Transformers architecture and can be utilized for text generation tasks. One of the first models of this type was GPT-2 published by Radford et al. (2019). This type is currently a big trend and is widely utilized for large language models (LLMs). One such model is LLaMA proposed by Touvron et al. (2023), which achieves state-of-the-art performance and is trained solely using publicly available data.
- **Encoder-decoder models**. Such models use both the encoder and decoder of the original Transformers architecture. Encoders are used for analyzing an input sequence and decoders for output generation. This type is a good choice for sequence-to-sequence tasks. An example that utilizes this architecture is BART (Bidirectional

and Auto-Regressive Transformers) presented by Lewis et al. (2020). Another popular model is T5 (**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer) presented by Raffel et al. (2020).

Pre-trained models use unsupervised data for pre-training. During pre-training encoder-decoder models (BART and T5) corrupt an input sequence and try to reconstruct it. Lewis et al. (2020) state that the pretraining of BART is carried out in two steps. In the first step, an input text is corrupted using one of the noising functions. In the second step, models learn how to reconstruct the original text from the corrupted version. Pre-training is performed by minimizing a reconstruction loss, which is calculated as the cross-entropy between the output of the decoder and the original document.

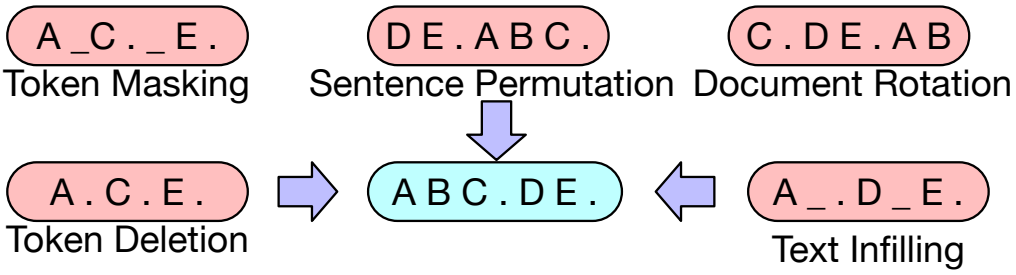

Figure 4: Noising functions allowed by BART (Lewis et al. 2020).

Models can use various noising functions for text corruption. BART, for example, allows the utilization of the following types of noising functions that are also demonstrated in Figure 4:

- **Token Deletion**: Random tokens are deleted from the input. The model must insert missing inputs.
- **Token Masking**: Random tokens are replaced with [MASK] elements. The model must replace [MASK] tokens with the correct tokens.
- **Text Infilling**: Random text spans with varying numbers of tokens are sampled and replaced with a single [MASK] token. The model learns to predict the number of missing tokens in each span.
- **Sentence Permutation**: Sentences within the text are shuffled randomly. The model learns to detect the correct order.
- **Document Rotation**: The document is rotated in such a way that it begins with one randomly chosen token. The model learns to identify the starting point of the document.

Pre-trained models can be easily fine-tuned for various sequence-to-sequence tasks. The challenge, however, is that the original versions of some models listed above are pre-trained exclusively on data in a single language (English) and can not be utilized for multilingual sequence-to-sequence tasks.

To address this challenge, it is possible to pre-train multilingual models capable of encoding and generating texts in different languages. Some of the models listed above

(BERT (Devlin et al. 2019)., BART (Lewis et al. 2020), and T5 (Raffel et al. 2020)) have their multilingual versions (e.g. mT5 presented by Xue et al. (2021) and mBART published by Yinhan Liu et al. (2020)). These models are pre-trained using multiple languages simultaneously, sharing vocabulary and model weights. This sharing facilitates transfer learning between languages due to the similarities among them. Especially, low-resource languages could benefit from this transfer learning.

Pre-training of multilingual models is conducted similarly to the original models, but for different languages simultaneously. To inform a model about the language of the input sequence and the expected language of the output text, multilingual models use language tokens. These tokens are simply placed at the beginning or end of input sequences.

2.3 Summarization task

Summarization is one of the sequence-to-sequence tasks that can be tackled by using pre-trained Transformers models. Summarization is a natural language processing task that aims to shorten longer texts while preserving their key information, important details, and overall meaning. It reduces long documents to one or a few sentences. Summarization can be used for faster information extraction, headline generation, abstract generation for research papers, preprocessing for text classification and information retrieval, and other applications.

As Allahyari and Safaei (2017) state, manual summarization done by humans is a time-consuming process that requires significant resources. Therefore, it is a perfect candidate for automation. However, automatic text summarization faces significant challenges since machines lack human knowledge and language capabilities, making it a complex and non-trivial task.

Pilault et al. (2020) note that automatic text summarization can be accomplished using two main techniques: extractive and abstractive. Extractive summarization techniques involve selecting a subset of words, phrases, or sentences from the input document and constructing a summary using them, without introducing new words or paraphrasing. The selected sentences should be concatenated in a manner that ensures readability, avoids redundancy, and maintains coherency.

One advantage of extractive summarization is that it relies only on the information present in the original text, without introducing any additional details. At the sentence level, extractive summarization can be seen as a binary classification problem, with the first class representing sentences to be included in the summary, and the second class containing sentences that should be ignored (Gialitsis, Pittaras, and Stamatopoulos 2019). A formal description of the task is as follows:

A given document D consisting of a sequence of N sentences $S = s_i, i \in N$*, is modified to a subset of M sentences (where* $M < N$*) using a classifier that predicts a class (0, 1) for each sentence (1 if a sentence should be included to the summary, 0 if not).*

Abstractive summarization involves creating summaries that may include expressions, sentences, or words not present in the original texts, while also capturing the meaning, key ideas, and elements of the source text. One advantage of abstractive summarization is its ability to generate summaries that resemble human-like writing, thereby enhancing readability (Pilault et al. 2020). Abstractive summarization can be viewed as a sequence-to-sequence problem. It takes as an input a sequence of tokens $x = [x_1; \ldots; x_m]$ and generates a target sequence $y = [y_1; \ldots; y_n]$, so that y contains key infromation from x ($y = argmax_y P(y|x)$) and $n < m$ (Chopra, Auli, and Rush 2016).

Table 1 shows an example of extractive and abstractive summaries.

Text	Coventry firm Travel de Courcey is to introduce the three buses in May next year, on its Park and Ride South route. The 38-seat buses will run between the Memorial Park in Kenilworth Road and the city centre using power points already installed by the council. A Travel de Courcey spokesman said the company had been looking to improve its vehicles, both environmentally and from a passenger perspective. The buses, Versa EV's, are provided by Optare plc of Leeds. Travel de Courcey has invested £400,000, the government's Green Bus Fund has invested £300,000 and Centro, which looks after public transport in the West Midlands, has contributed £100,000. Mike de Courcey, from the bus firm, said when it heard about the Green Bus Fund it seemed a good opportunity for the firm. "The electric buses are ideal for urban driving where the vehicle is stopping and starting," he said.
Extractive summary	Coventry firm Travel de Courcey is to introduce the three buses in May next year, on its Park and Ride South route. The 38-seat buses will run between the Memorial Park in Kenilworth Road and the city centre using power points already installed by the council.
Abstractive summary	Electric buses will soon be running on the roads in Coventry.

Table 1: Examples of extractive and abstractive summaries.

2.4 Zero-shot and few-shot multilingual abstractive summarization

Multilingual abstractive summarization refers to the generation of summaries for texts written in different languages. Training a single model that performs well with various languages can be challenging. Such models need to encode and generate texts in different languages, which can vary significantly in terms of alphabets, vocabulary, grammar, syntax, and idiomatic expressions.

Multilingual summarization can be categorized into two types: intralingual and cross-lingual. Intralingual multilingual summarization models can handle different languages, but the input text and generated summaries are in the same language. Cross-lingual multilingual summarization, on the other hand, can generate summaries in languages different from the language of the input texts.

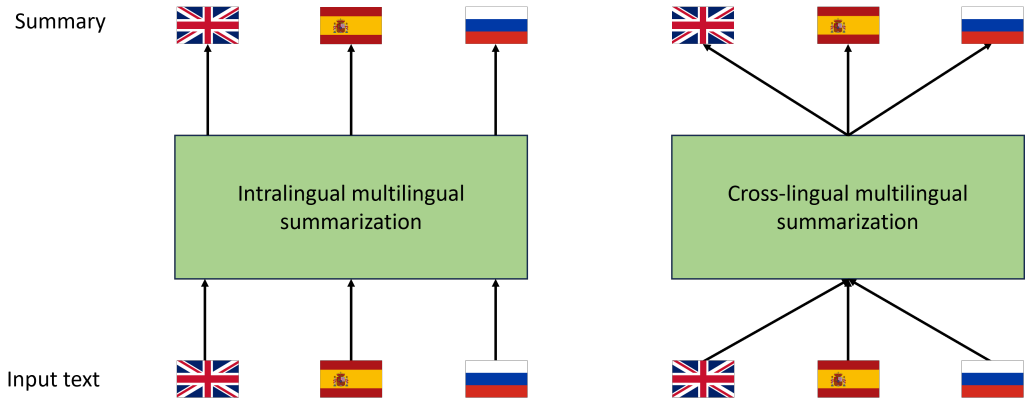

Figure 5: Two types of multilingual summarization models. Left: intralingual model; Right: cross-lingual model.

Training models for multilingual summarization requires a substantial amount of supervised training data, especially for different directions of cross-lingual scenarios. One possible solution to this problem is training a single model on multilingual data, enabling it to learn relationships between languages. Such models benefit from transfer learning, as languages within the same family share similarities in summarization processes. As the process of summarization is similar in different languages, training a single model for multiple languages can improve results and facilitate summarization in those languages.

Another improvement can be achieved by utilizing pre-trained multilingual models that have been pre-trained on extensive amounts of unsupervised multilingual data. Although these models cannot be directly used for any task, they possess knowledge about language relationships and usually have a large vocabulary. To employ pre-trained multilingual models for multilingual summarization, they need to be fine-tuned using supervised data. Fine-tuning a model in one language enables summarization in other languages, as the model has acquired knowledge about those languages from the pre-training phase.

Zero-shot multilingual summarization refers to training a model using data in certain languages and evaluating it using languages that complete one of the following conditions:

- The input language or the output language is unseen during the fine-tuning process.
- The input language and the output language are available during the fine-tuning process, but not in such a combination.

In some cases, achieving good results in zero-shot scenarios can be challenging. In such situations, few-shot learning can help improve performance. Few-shot learning differs from zero-shot learning in that it utilizes a small amount of supervised data. An advantage of few-shot learning is that it is relatively easy to prepare a small amount of data. By adapting a pre-trained model quickly, few-shot learning can enhance results compared to zero-shot learning. The amount of data required for few-shot learning depends on the task and dataset.

3 Related work

This chapter discusses various methods that can enhance the performance of multilingual summarization in zero-shot and few-shot scenarios. To achieve better results in zero-shot and few-shot multilingual summarization, the methods should have at least one of the following objectives:

- **Maximizing transfer learning and minimizing catastrophic forgetting.** Maximize the benefits of utilizing a multilingual pre-trained model and minimize the consequences of *catastrophic forgetting*. This term was introduced by Ratcliff (1990) and French (1999). Catastrophic forgetting is a definition used in situations when pre-trained models are fine-tuned in such a way that they are adapted to a new task or new type of data so that they can not perform well on the tasks and data that were used previously during pretraining. In the case of the multilingual model, fine-tuning the multilingual model using only one language can make this model specific only to this language. As multilingual models are pre-trained with different languages and benefit from the transfer learning between languages, it is helpful to try to overcome the problem of catastrophic forgetting to be able to keep this multilingual knowledge.
- **Enforcing language-independent encoder outputs.** Make the output of the encoder less language-specific. Even though all languages use the same encoder and share encoder parameter weights, the encoder representation still contains language-specific characteristics, which leads to poor zero-shot results according to Yang et al. (2018). Ideally, it should be impossible to determine which language the encoder output is related to, making it similar to an interlingua. This property enables the use of input languages that were unavailable during fine-tuning and improves the performance of previously unseen combinations of input and output languages.
- **Enforcing correct target language.** Ensure that the output is in the expected language. Models with multilingual generation capabilities must be conditioned on the desired target language at inference time. This constraint is often integrated into the model by dedicated target language tokens or specific language prompts (Vu et al. 2022). In practice under few- or zero-shot conditions, this is not always sufficient, leading to the generation of summaries in incorrect languages.

3.1 Maximizing transfer learning and minimizing catastrophic forgetting

3.1.1 Partial model fine-tuning

A possible solution to mitigate catastrophic forgetting is partial fine-tuning of pre-trained models. Maurya et al. (2021) and Chi et al. (2020) demonstrated that updating the weights of only selected parts of pre-trained models during fine-tuning leads to better results. Updating all weights of the model during fine-tuning with specific data, such as in one particular language, adapts all weights to this data and makes models less multilingual. All weights of models earlier pre-trained using multilingual data become language-specific and forget the knowledge of other languages. It can lead to the *off-target generation* problem described by Zhang et al. (2020). The authors originally called it *off-target translation* as they use it for machine translation. It means that the model consistently generates output texts in the wrong output language, regardless of a target language token that is supposed to force an expected output language.

Fine-tuning only specific parts of pre-trained models and freezing others can help the frozen parts of the models retain their knowledge of different languages obtained during pre-training. Fine-tuning pre-trained models for a summarization task should aim to learn how to summarize texts effectively while preserving knowledge of different languages. For this purpose, it is beneficial to identify a subset of model components that should be updated and those that should remain unchanged.

Maurya et al. (2021) and Chi et al. (2020) conducted zero-shot experiments on various NLP generation tasks, specifically focusing on multilingual abstractive summarization. Maurya et al. (2021) fine-tuned a pre-trained mBART model and utilized intralingua data from the WikiLingua dataset (Ladhak et al. 2020). The model was trained with English data and subsequently evaluated using Hindi and Japanese data in zero-shot scenarios. Chi et al. (2020) employed a multilingual Transformer model consisting of 10 encoder layers and 6 decoder layers. This model was fine-tuned using intralingual data from the Gigagword summarization dataset (Napoles, Gormley, and Van Durme 2012). Fine-tuning was performed on English data and evaluation in zero-shot scenarios using French and Chinese data. In addition to these experiments, the authors of both papers also conducted cross-lingual experiments for other generation tasks, including question generation and headline generation.

In both papers, the authors observed that fine-tuning a multilingual model with data in just one language (e.g., English) results in catastrophic forgetting, leading to the off-target generation problem. The models fine-tuned by the authors perform well when encountering unseen input languages but struggle when dealing with output text languages that were not part of the fine-tuning process. Consequently, the model's decoder forgets other languages and consistently generates summaries in English.

To address this issue, the authors apply partial fine-tuning and propose freezing all parameters of the decoder layers and word embeddings and tuning only weights in encoder

layers during fine-tuning with English data (Maurya et al. 2021). This approach leads to improved results, although it does not completely resolve the problem. The generated outputs are no longer entirely in English but may still include some phrases in English.

Li et al. (2021) conducted multilingual speech-to-text translation experiments, employing an encoder from the wav2vec 2.0 model for speech recognition and a decoder from the mBART model for output text generation. The authors examined how freezing model parameters and selectively tuning certain parts of the model could enhance its performance in zero-shot scenarios, including unseen languages on both the source and output sides, as well as unseen language pair directions. They found out that fine-tuning the encoder together with the normalization layers and encoder attention within the decoder layers leads to benefits in zero-shot scenarios. Furthermore, in some cases, it also resulted in improved performance to freeze all parameters of the encoder except for the normalization layers and self-attention layers within the encoder layers.

3.1.2 Language adapters fine-tuning

Another option to increase transfer learning and minimize catastrophic forgetting is to use language-specific or domain-specific adapters. Bapna and Firat (2019) were the first authors who proposed the utilization of language-specific adapters for neural machine translation. In their work, they employ adapters for adapting pre-trained models to new domains and languages. Such adapters are added to pre-trained models and during fine-tuning only the weights of these adapters are updated and the rest of the model stays unchanged.

Language pair adapters. Adapters proposed by Bapna and Firat (2019) are placed at the end of every layer and can be used in both encoder and decoder layers. In encoder layers, they enhance the encoding of input languages. In decoder layers, they improve the generation of higher-quality output text by considering specific language details. Figure 6 demonstrates the architecture of adapters proposed by Bapna and Firat (2019). Firstly, there is a normalization layer, followed by a down-projection of vectors to a bottleneck dimension and a rectified linear unit (RELU) activation function. The output is then up-projected and connected with the original value through a residual connection.

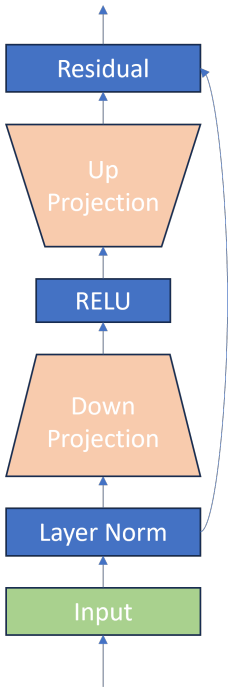

Figure 6: Adapter architecture proposed by Bapna and Firat (2019).

Adapters proposed by Bapna and Firat (2019) are bilingual, language-pair-specific adapters, meaning one adapter for each language pair. While this approach can lead to improved adaptation, it also has a few drawbacks. It necessitates training adapters for every language pair, resulting in $O(N^2)$ pairs. Training can be time-inefficient and requires supervised data for all language pairs, making this solution impractical for zero-shot scenarios.

Monilingual language adapters. Philip et al. (2020) proposed the use of adapters with the same architecture, but as monolingual, language-specific adapters. Adapters in encoder layers are specific to the input text language, and adapters in decoder layers correspond to the expected output languages. This approach requires fewer adapters $O(2 * N)$ and allows for the combination of any input and output languages seen in training including unseen pairs.

3.2 Enforcing language-independent encoder outputs

3.2.1 Adversarial language classifier

The problem of language-independent encoder representation has been analyzed by multiple authors in the field of neural machine translation. Yang et al. (2018) enabled translation between languages using a model trained solely on monolingual unsupervised data. Arivazhagan et al. (2019) utilized supervised cross-lingual English-centred data for training but conducted zero-shot experiments for unseen pairs of languages.

In both papers, the authors discovered that the encoder representation in encoders trained across multiple languages remains language-specific, even though they share all weights and are trained simultaneously. The authors demonstrate that making the encoder representation less language-specific and more similar to an interlingua significantly enhances results in zero-shot scenarios.

To achieve this objective, the authors utilize an adversarial loss. They construct a language classification discriminator on the top of the encoder, capable of predicting the language of the input text. The model is trained in such a way that the discriminator attempts to minimize a classification loss, while simultaneously the encoder should be updated to maximize the classification loss.

In my thesis, I employ an updated solution proposed by D. Liu and Niehues (2022). The authors use a language classification discriminator on the token level. The training is conducted in two alternating steps, which help prevent the coadaptation of parameters and contribute to a more stable training process:

- **Step 1.** Encoder output classification step. In this step, only the language classification discriminator is trained and updated. It minimizes the cross-entropy loss between predicted languages and the expected ones on the token level. Equation 2 is applied in this step.

$$\begin{aligned} &\mathcal{L}_{\text{classifier}} = -\sum_{l=1}^{\#Languages} y_l \log(p_l), \text{ where} \\ &y_l = 1 \text{ if } l \text{ is the expected language of the token,} \\ &y_l = 0 \text{ otherwise,} \\ &p_l \text{ is the predicted probability that a token belongs to the language } l \end{aligned} \tag{2}$$

- **Step 2.** In the second step, the objective is to fool the language classification discriminator. This could be achieved by updating the encoder weights using a negative loss from the first step 2. However, the issue is that this loss can be very small if the classifier performs well, leading to minimal updates of the encoder weights. This is why the authors propose an adversarial classification loss, calculated as shown in equation 3.

$$\mathcal{L}_{\text{adversarial}} = \sum_{l=1}^{\#Languages} y_l \log(1 - p_l), \tag{3}$$

For better stability, a summarization task loss is added to the adversarial loss. Finally, equation 4 is employed for training in the second step.

$$\mathcal{L}_{\text{encoder\_decoder}} = \mathcal{L}_{\text{summarization}} + \mathcal{L}_{\text{adversarial}} \tag{4}$$

3.2.2 Removing residual connections

Another possible approach that aids in achieving a more language-independent encoder representation is removing residual connections. This concept was introduced by D. Liu, Niehues, et al. (2021). In this paper, the authors conducted neural machine translation experiments by training models using English-centered data and evaluating zero-shot scenarios for unseen language pairs. They observed that the order of the encoder representation corresponds significantly to the order of the encoder input and that this positional information is also language-specific. The issue lies in the fact that it hinders the generation of a less language-specific encoder representation, even when each token's encoder representation is language-independent. While positional information plays an important role in some natural language processing tasks, such as part-of-speech tagging, the authors demonstrate that it leads to poorer results in zero-shot scenarios in machine translation.

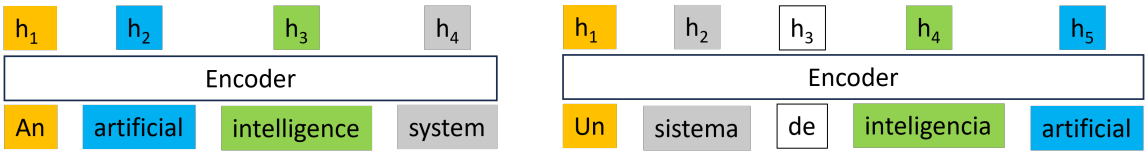

Figure 7: Different encoder representation orders in English and Spanish.

Figure 7 illustrates an example of how two phrases with the same meaning in English and Spanish are encoded. The encoder output representations have different orders, and a Spanish phrase has one extra token. Even if all encoder output representation tokens are language-independent, it is still possible to determine the language of the input sequence, as different languages may have specific grammar rules for constructing sentences.

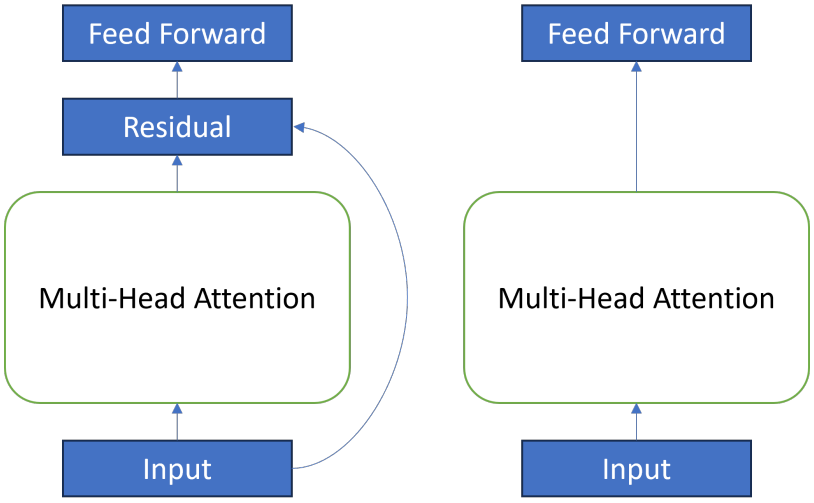

Figure 8: Left: standard encoder layer implementation; Right: with removed residual connection.

The authors D. Liu, Niehues, et al. (2021) propose a technique aimed at reducing positional information in the encoder output representation. They recommend removing a residual connection that links an input and an output of a self-attention mechanism in a single encoder layer (as shown in Figure 8). Residual connections are valuable for improving the gradient flow from losses to lower layers and mitigating the vanishing gradient problem. They also maintain a one-to-one correspondence between the encoder input sequence and the encoder output representation. The removal of such a residual connection in one encoder layer degrades this correspondence. Since I am using a pre-trained multilingual model in my thesis, the vanishing gradient problem does not significantly impact training because the lower layers are already well pre-trained and capable of encoding input sequences.

When selecting the encoder layer in which to implement this solution, the authors decide that it is better to do so in the middle layer. They remove the third and fifth layers of the models with 5 and 8 encoder layers, respectively.

The authors conducted experiments with various languages, demonstrating that their implementation leads to improved results in zero-shot scenarios, closely approaching the performance of machine translation using a pivot language on average. Their experiments show that, when translating between languages of the same family (Germanic or Romance), the results are even better compared to using a pivot language.

4 Proposed approaches

This chapter presents various approaches that I propose and that can be applied during multilingual summarization model training to improve its performance. All of these approaches are evaluated in Chapters 6 and 7 through experiments.

4.1 Key-query fine-tuning

A new approach proposed in this section, fine-tuning only keys and queries in multi-head attentions, aims to maximize transfer learning and minimize catastrophic forgetting during the fine-tuning of pre-trained multilingual models. Section 3.1 describes that fine-tuning only a part of the model parameters can increase gains from the usage of pre-trained multilingual models and improve zero-shot results. Maurya et al. (2021) and Chi et al. (2020) proposed to update only the weights of encoder layers during fine-tuning. Li et al. (2021) suggests updating both encoder and decoder weights during fine-tuning, but not all of them. Within the encoder layers, only normalization layers and self-attention layers are updated. Within the decoder layers, only the normalization layers and encoder attention layers are updated.

Additionally, I propose exploring the possibility of finding an intermediate solution that requires fine-tuning fewer parameters while improving performance simultaneously. Pre-trained multilingual sequence-to-sequence models can reconstruct corrupted input sequences in different languages. It means they can encode and analyze input texts and generate coherent output texts. The distinction between the reconstruction task and the summarization task lies in how they determine which parts of the input sequence are relevant for output generation. Concerning text generation, both tasks operate similarly.

The goal is to maximize transfer learning from the pre-trained model regarding output generation in different languages and to learn to select the most important parts of input sequences for summarization. To achieve this, it could be beneficial to update only the weights of parts of the models responsible for such selection, leaving the weights of other components unchanged. Updating the weights of other components can make them more language-specific and reduce transfer learning. As described in Section 2.1, the mechanism responsible for selecting the most important parts of sequences during encoding and decoding is multi-head attention. To leave text generation unchanged and to analyze input sequences, only two of the three multi-head attentions should be fine-tuned: self-attentions in encoder layers and encoder-attentions in decoder layers. Self-attentions in decoder layers should not be updated as they are responsible for the generated output text.

Multi-head attentions receive three vectors (values, queries, and keys) as input, and apply linear projections to them initially (see Section 2.1). I propose fine-tuning only the weights of linear projections applied to queries and keys. The reason for this is that, through queries and keys, the importance of every position is calculated, helping in the selection of parts of sequences important for generated summaries. Updating weights of linear projections applied to values could make their weights more language-specific, leading to direct language-specific transformation. In contrast, updating queries and keys does not directly manipulate values but only decides on the importance of values at every position. This approach could lead to a more focused consideration of the input text structure and better utilization of transfer learning from the pre-trained multilingual model.

4.2 Improved adversarial language classifier

In my thesis, I will conduct experiments that enable the training of a cross-lingual model using only monolingual data. For this purpose, I will try to use an adversarial language classifier that was presented in Section 3.2.1. Such adversarial language classifiers aim to make encoder outputs language-independent. In this section, I provide an updated version of the adversarial language classifier that could create even more language-independent encoder outputs.

In the best case, after applying an adversarial language classifier, it should not be possible to differentiate between encoder output representations for inputs in different languages. When training a model utilizing N input languages, language classification probabilities should have a uniform distribution with a value of $1/N$ for inputs in every language, as shown in Diagram "d) Expected distribution" in Figure 9. In this case, the classifier really is unable to distinguish input languages. The drawback of the original solution described in Section 3.2.1 is that it could happen that it does not lead to equal probabilities between all languages. Potentially, it could have two problems:

- **Too big changes.** It could penalize the output of the actual language so much that it might map this actual language to another language. Applying this approach would simply map some languages to incorrect labels, leading to a significant confusion of the classifier, but not necessarily resulting in equal probabilities between all languages. Diagram "b) Too big changes" in Figure 9 shows the potential outcome. The probabilities are more equal compared to the original probabilities presented in Diagram "a) Original probabilities" in Figure 9 but are manipulated too extensively. This situation could especially occur if training is performed using unbalanced data.

- **Too small changes.** Another problem that can occur is that penalizing is not sufficient and changes resulting from applying an adversarial language classifier are too small. Diagram "c) Too small changes" in Figure 9 illustrates this situation. Although it performs changes toward a uniform distribution, it may not be sufficient for achieving a language-independent encoder representation.

I propose using a modified adversarial classification loss that considers the drawbacks listed above and could make the encoder output representation less language-specific. I suggest employing Kullback–Leibler divergence (see 5) instead of the original equation 3, which enforces more equal probabilities for all languages. This modification would not result in too big changes, as it leads to making probabilities equal to $1/N$ and not to zero. Another advantage compared to the original equation is that the use of Kullback–Leibler divergence does not only penalize the output of the actual language but also actively encourages outputs of other languages. Performing only the penalization of actual languages updates the weights of model components in such a way that it makes this actual language less probable, but at the same time, it does not force the output of other languages. This can lead to the problem of too small changes.

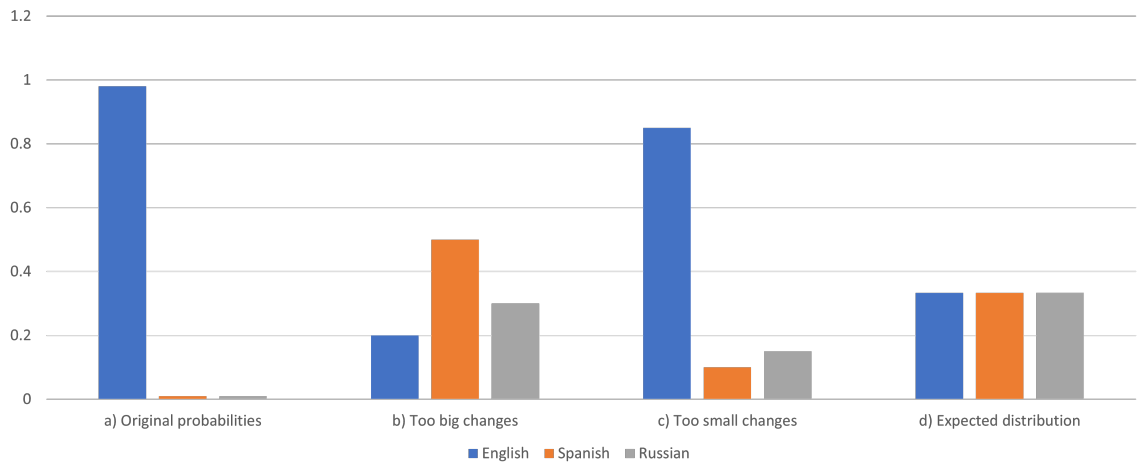

Figure 9: Encoder output representation classification results for data in English.

$$\mathcal{L}_{\text{adversarial kldivloss}} = D_{kl}(encoder\ output\ classification\ ||\ equal\ probabilities),$$
where $equal\ probabilities$ is a uniform distribution with the value
$$\frac{1}{\#Languages}, \text{ e.g. } [1/3, 1/3, 1/3] \text{ for 3 languages} \tag{5}$$

As an adversarial language classifier works at the token level, it removes language-specific information from each token but not from the entire encoder output representation. It is still possible that the encoder output representation contains language-specific information represented by the structure of the output sequence. Therefore, it can be helpful to additionally remove position information from the encoder output representation. For this purpose, I apply the removal of residual connections from Section 3.2.2 to the model and train it again in combination with an adversarial loss. The combination of these two approaches should make the output sequence less language-specific and improve results.

4.3 Adapters for enforcing target language

In this section, I propose approaches that aim to improve control over the target language. In multilingual models, the output language is typically triggered using a language token during decoding. However, in many cases, this mechanism is not sufficient as the signal from the target token can be weak and the output language also depends significantly on the language-specific characteristics of the encoder output. As a potential solution for more effectively forcing the generation of summaries in the desired languages, I propose to use language-specific adapters that were presented in Section 3.1.2.

For my needs, I adopt monolingual adapters proposed by Philip et al. (2020). For stronger enforcement of the output language, I apply a language adapter corresponding to the output language in both the encoder and decoder layers. This approach differs from the original solution proposed by the authors.

In the original papers, authors initially pre-train models for a machine translation task. Afterward, they freeze all parameters, add adapters, and continue training only them further. In my thesis, I will train all model parameters and adapters together because my objective is to use them as a mechanism to enforce the output to be in the expected language.

I also suggest another solution, that helps to adapt an encoder output representation to the expected output language. For this purpose, I propose the use of language-specific adapters at the token level of the encoder output representation. These adapters consist of simple linear projections with the same dimensions, combined with dropout and a rectified linear unit (RELU) activation function. Figure 10 illustrates its architecture. The combination of these adapters with language tokens should make it easier to ensure that a summary is generated in the expected language.

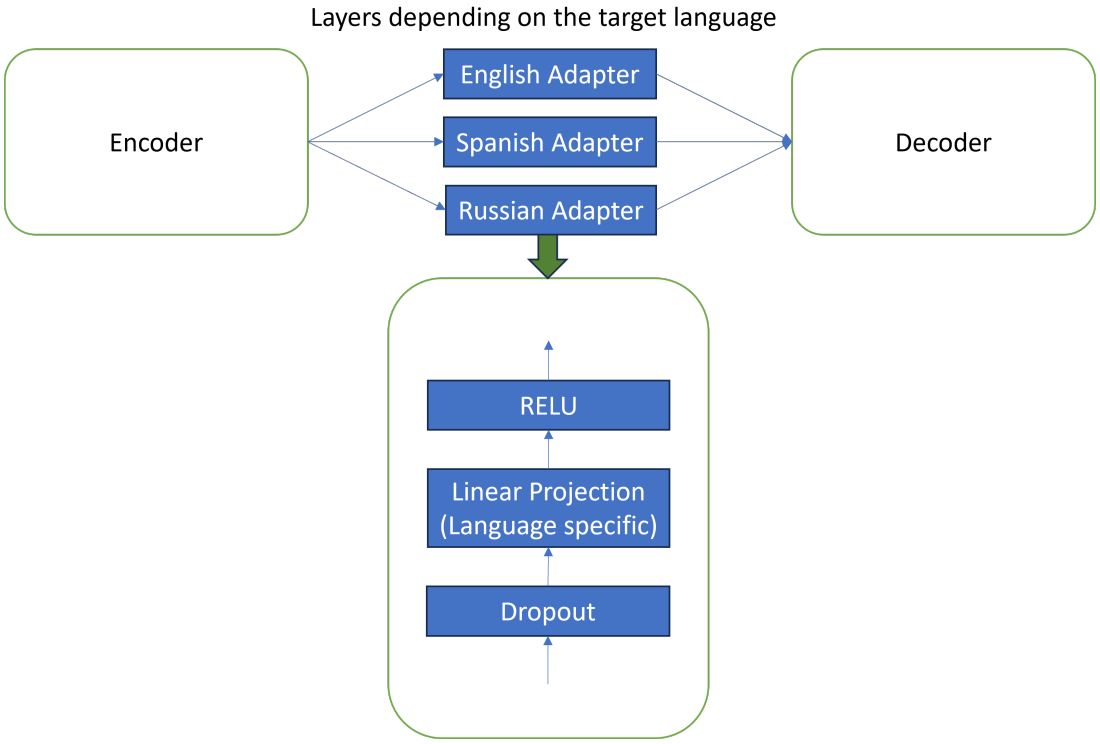

Figure 10: Proposed encoder output adapter.

5 Experimental setup

5.1 Baselines

5.1.1 Zero-shot setup

Baseline zero-shot results demonstrate the performance of zero-shot cases conducted using models trained without any advanced approaches. These baseline results are used for comparisons and should be improved in further experiments employing different approaches.

To obtain zero-shot results, a model is first fine-tuned using a subset of language pairs. Subsequently, in zero-shot cases, language pairs are evaluated that did not participate in the fine-tuning process. When selecting language pairs for zero-shot cases, different combinations of languages should be considered:

1. A language pair contains an input or output language that was unseen during fine-tuning.

2. A language pair consists of languages, both of which were available during fine-tuning but not in such a combination.

5.1.2 Translation-based solution

One possible solution for multilingual summarization is to train a single model using data in only one language, such as English, as it often has the most training data, and employ other neural machine translation models for data pre- and post-processing. In this case, firstly, I translate input texts from other languages to English. Secondly, I generate summaries using a summarization model trained with English data. Finally, if required, I translate the generated summary in English into another language. Figure 11 illustrates how this scenario works.

An advantage of this approach is that it requires less supervised data and can be applied to many languages. Modern neural machine translation models can produce highly qualitative translations to and from English, especially for texts in high-resource languages. In my thesis, I use two models and compare their results: mBART50 (presented in Section 5.3.2) and NLLB (presented in Section 5.3.3).

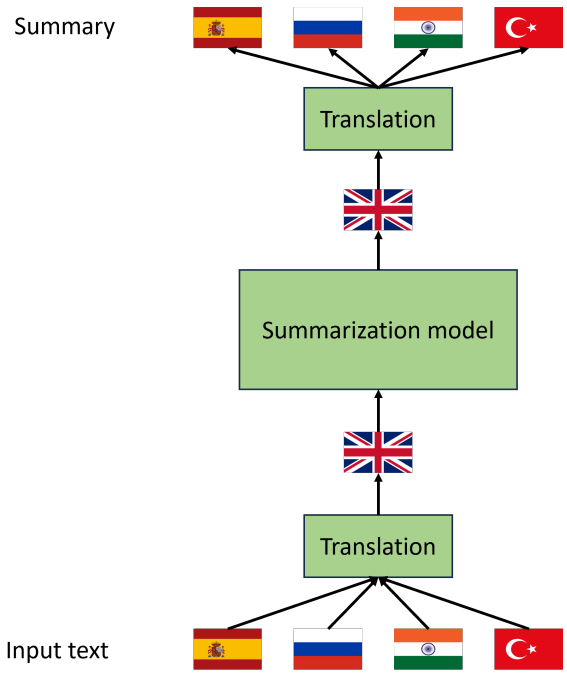

Figure 11: Multilingual summarization using a summarization model trained with English data and neural machine translation models.

5.2 Few-shot setup

A model that is already capable of generating summaries in one language should require less data to be adapted to other languages, benefiting from transfer learning between different languages. By performing few-shot experiments, it is possible to analyze the amount of supervised data needed to adapt models to new languages unseen during fine-tuning and achieve good results for them.

For this purpose, I take fine-tuned baseline models that I trained earlier for other language pairs and further fine-tune them with a limited amount of data in the desired languages. Evaluations are performed using the following amounts of data: 10, 100, 1000, and 10000 (if there is enough data for a language pair). The results should demonstrate how easily models can be adapted to other languages and how much data is needed to obtain good results.

5.3 Utilized pre-trained models

5.3.1 mBART

For completing a summarization task in my experiments, I utilize a multilingual version of BART called mBART proposed by Yinhan Liu et al. (2020). This model pre-trains a complete autoregressive sequence-to-sequence model by reconstructing original texts corrupted by two denoising functions: token infilling and sentence permutation. Token infilling is done by masking 35% of the words in each instance by random sampling spans of text. Sentence permutation is performed in each instance. Figure 12 shows an example of pretraining and denoising functions. The mBART model utilizes large-scale intralingual corpora across many languages for pre-training. Pre-training is done once for all languages at the same time. A subset of 25 languages, CC25, extracted from the Common Crawl (CC) (Wenzek et al. 2020) is used for pre-training. This subset contains languages from different language families and with different amounts of data. For tokenization a sentence-piece model (SPM) model (Kudo and Richardson 2018) is utilized that contains 250000 tokens.

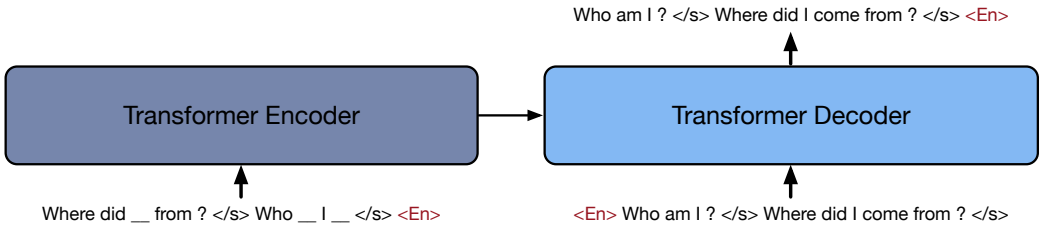

Figure 12: mBART pre-training (Yinhan Liu et al. 2020).

mBART architecture contains 12 encoder layers and 12 decoder layers with about 680M parameters. All numbers of parameters are described in Table 2.

# parameters	680M
# encoder layers	12
# decoder layers	12
h (# heads)	16
d_{model}	1024
$d_{embeddings}$	1024
d_k	64
d_v	64
d_q	64
$d_{FeedForward}$	4096
vocabulary (# tokens)	250000
# supported languages	25

Table 2: mBART parameters.

To control input and output languages, language tokens are used at three positions:

- At the end of the encoder input.
- At the end of the decoder output.
- At the beginning of the decoder input.

Setting a language token at the beginning of the decoder input during interference is supposed to force output text to be in the expected language.

5.3.2 mBART50

mBART50 is an extended version of mBART proposed by Yinhan Liu et al. (2020), supporting 50 languages instead of the original 25. I utilize two fine-tuned mBART50 models for translation purposes. The first model is many-to-one, translating input texts from various languages to English. The second model is one-to-many, translating generated summaries from English to other languages. For the translation process, I employ the EasyNMT library (`https://github.com/UKPLab/EasyNMT`), which offers wrappers for different machine translation models, including mBART50 models.

5.3.3 No Language Left Behind - NLLB

Another model that I use in my thesis for experiments is No Language Left Behind (NLLB) presented by Team et al. (2022). It is a single model capable of performing many-to-many cross-lingual translations from and to 200 different languages, including many low-resource languages. The authors of NLLB demonstrate that their model has significant improvements compared to previous state-of-the-art models. There are different versions of NLLB models. In my thesis, I use the smallest one, NLLB-200-Distilled, with 600 million parameters, which is similar to the mBART50 model. For the translation process, I utilize HuggingFace integration (`https://huggingface.co/docs/transformers/model_doc/nllb`).

5.4 Hyperparameters

I conduct experiments using the open-source sequence modeling toolkit FAIRSEQ, as presented by Ott et al. (2019). This implementation allows the training of models for different generation tasks, such as translation and summarization. For my experiments, I utilize "translation_multi_simple_epoch" task that is designed for translation experiments. I adapt this task to suit the requirements of the multilingual summarization task. All modifications and implementations can be accessed in my repository, which is forked from the original project: `https://github.com/vladsolovyev/fairseq_summarization/tree/main/summarization_scripts`.

I conduct summarization experiments using an mBART model, as described in Subsection 5.3.1. I utilize the pre-trained model "mbart.CC25", downloaded from `https://github.com/facebookresearch/fairseq/blob/main/examples/mbart/README.md`.

Table 3 provides an overview of the hyperparameters used during the fine-tuning of models and inference. Further settings and all reproducible scripts can be found in my repository, as provided above.

optimizer	adam
adam eps	1e-8
adam betas	(0.9, 0.999)
learning rate scheduler	polynomial decay
learning rate	2e-5
end learning rate	5e-9
weight decay	0.01
dropout	0.1
attention dropout	0.1
beam size	5
length penalty intralingual	0.6
length penalty cross-lingual	1.0
min length intralingual	0
min length cross-lingual	10
max length	100
no repeat ngram size	3

Table 3: Experiments training and generation hyperparameters.

5.5 Evaluation

To evaluate the performance of multilingual summarization models, automatic evaluation metrics can be used. These metrics offer quick feedback regarding the quality of generated summaries in different languages. For effective multilingual summarization, it is important that the generated summaries contain all key information from the input texts, and the output text language corresponds to the expected language.

5.5.1 Rouge metrics

Lin (2004) proposed a new metric called ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation. ROUGE-N metrics count how many n-gram and word sequences from reference summaries appear in generated summaries. Normally, ROUGE-1 and ROUGE-2 are used, which are calculated using unigrams and bigrams, respectively. Equation 6 illustrates how ROUGE-N metrics are calculated.

$$Rouge_N = \frac{\sum_{S \in References} \sum_{gram_n \in S} CountMatch(gram_n)}{\sum_{S \in References} \sum_{gram_n \in S} gram_n} \tag{6}$$

Candidate: She learns computer science at a European university. **Reference:** She studies computer science at the university in Europe. **Candidate unigrams:** "She", "learns", "computer", "science", "at", "a", "European", "university" **Reference unigrams:** "She", "studies", "computer", "science", "at", "the", "university", "in", "Europe" **Candidate bigrams:** "She learns", "learns computer", "computer science", "science at", "at a", "a European", "European university" **Reference bigrams:** "She studies", "studies computer", "computer science", "science at", "at the", "the university", "university in", "in Europe"
$Rouge_1 = \frac{5}{9} \quad Rouge_2 = \frac{2}{8}$

Table 4: An example of Rouge metrics calculation.

Example 4 demonstrates how to create reference and candidate unigrams and bigrams and calculate ROUGE-1 and ROUGE-2 metrics.

A disadvantage of ROUGE-N metrics is that they do not consider the order of appearing N-grams. "A professor asked students." and "Students asked a professor." sentences would have the same ROUGE-1 metric values, even though they have different meanings. ROUGE-L addresses this drawback. ROUGE-L calculates the longest matching sequence of words using the longest common subsequence (LCS) approach. It identifies a common subsequence in two summaries with the maximum length.

A sequence z = $[z_1; \ldots; z_n]$ is considered a subsequence of another sequence x = $[x_1; \ldots; x_n]$ if there exists a strictly increasing sequence $[i_1; \ldots; i_k]$ of indices in x such that for all j = 1; ...; k, we have $x_i j = z_j$ (Lin 2004). LCS considers the order of the n-grams but does not require consecutive matches. This allows it to calculate the longest common n-grams without the need to predefine the length of the n-grams. To prevent longer summaries from receiving higher values disproportionately, the F-measure is applied. To estimate the overlap between a reference summary X of length m and a candidate summary Y of length n, the following equations for recall, precision, and F-measure are calculated. 5 provides formulas that are utilized for the calculation of ROUGE-LCS metrics. 6 gives an example of such a calculation.

$$R_{lcs} = \frac{LCS(X,Y)}{m} \quad P_{lcs} = \frac{LCS(X,Y)}{n}$$

$$\beta = \frac{P_{lcs}}{R_{lcs}} \quad F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

Table 5: Rouge-LCS metrics formulas.

Reference: A professor asks students.
Candidate 1: A professor asked students.

$$Rouge_1 = \frac{3}{4} \quad Rouge_{lcs} = \frac{3}{4}$$

Candidate 2: Students asked a professor.

$$Rouge_1 = \frac{3}{4} \quad Rouge_{lcs} = \frac{2}{4}$$

Table 6: An example of Rouge-LCS metrics calculation.

A disadvantage of the ROUGE-L score is that it considers only the longest common subsequence, and other shorter common subsequences do not have an influence on the final metric value.

5.5.2 BERTScore

Many metrics, including ROUGE, have a drawback in that they only work with exact n-gram overlaps and do not consider the semantic meaning of words and phrases. In this case, two synonyms have the same impact on the final score as two entirely different words. Additionally, the context in which n-grams appear does not play a role, making the evaluation inaccurate. For example, consider a reference sentence "A professor asked students," with two candidate sentences: "People in the class got a question from the lecturer," and "Students said hello to the professor." The ROUGE-1 metric assigns a higher value to the second candidate sentence than the first one, even though the meaning of the first candidate sentence is much closer to the reference sentence.

Zhang* et al. (2020) introduced a neural metric called BERTScore, which addresses this issue and enables judgments that are closer to human evaluation. When provided with a reference sentence $x = \langle x_1, \ldots, x_k \rangle$ and a candidate sentence $\hat{x} = \langle \hat{x}_1, \ldots, \hat{x}_l \rangle$, BERTScore utilizes contextual embeddings and computes token similarities by calculating pairwise cosine similarity between tokens, optionally applying inverse document frequency scores. Figure 13 illustrates an example of BERTScore calculation.

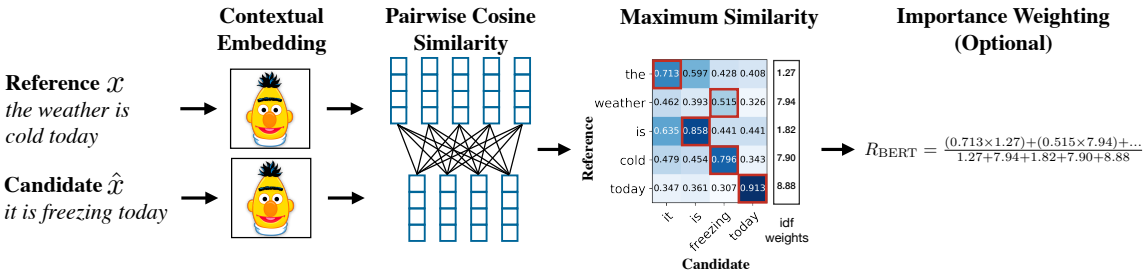

Figure 13: An example of BERTScore calculation (Zhang* et al. 2020).

In comparison to word embeddings like GloVe (Pennington, Socher, and Manning 2014) and Word2Vec (Mikolov et al. 2013), contextual embeddings can have distinct vector representations for the same word in various texts. To obtain such contextual embeddings, Transformer encoders can be employed, such as BERT models. Through their application, input sequences can be encoded, and each input token obtains its contextual representation. This thesis utilizes a multilingual version of BERT ("bert-base-multilingual-cased") since generated output summaries may be in different languages.

There are three types of BERTScore: recall, precision, and the F1 measure. For recall calculation, the contextual embedding of each token in x is compared to the contextual embedding of every token in $\hat{x}$. For precision calculation, the contextual embedding of each token in $\hat{x}$ is compared to the contextual embedding of every token in x. Afterward, a greedy matching strategy is applied to maximize the matching similarity score. This involves matching the contextual embedding of each token to the most similar contextual embedding of a token in the other sentence. The values of precision and recall are then combined to calculate the F1 measure.

For a candidate $\hat{x}$ and a reference x, the values of recall, precision, and F1 scores are determined using the following equations:

$$R_{BERT} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \tag{7}$$

$$P_{BERT} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \tag{8}$$

$$F_{BERT} = 2\frac{R_{BERT} \cdot P_{BERT}}{R_{BERT} + P_{BERT}} \tag{9}$$

A disadvantage of using the BERTScore metric with a multilingual BERT model is that the same words in different languages often have very similar semantic representations. This can result in situations where BERTScore values are high, even though the generated output summaries are in unexpected languages.

5.5.3 Language identification (LangID)

For the evaluation of multilingual summarization models, it is useful to identify the output languages. This helps in determining whether the models generate output summaries in the expected languages or not. To achieve this, the Language Identification (LangID) task can be utilized.

Lui and Baldwin (2012) introduced a language identification tool called *langid.py* to perform this task. The *langid.py* tool was trained using data from five different domains and supports 97 languages. As a classifier, it uses the naive Bayes classifier algorithm. Given a text and a list of possible languages as input, it returns the probabilities of each language being present in the input text.

6 Intralingual experiments

In this chapter, the results of intralingual experiments are presented. Intralingual multilingual models are models that can encode and generate texts in different languages, but the input and output languages are always identical. In my experiments, I work with two types of models:

1. models trained using only monolingual English data (Figure 14)
2. models trained using monolingual English, Spanish, and Russian data (Figure 15)

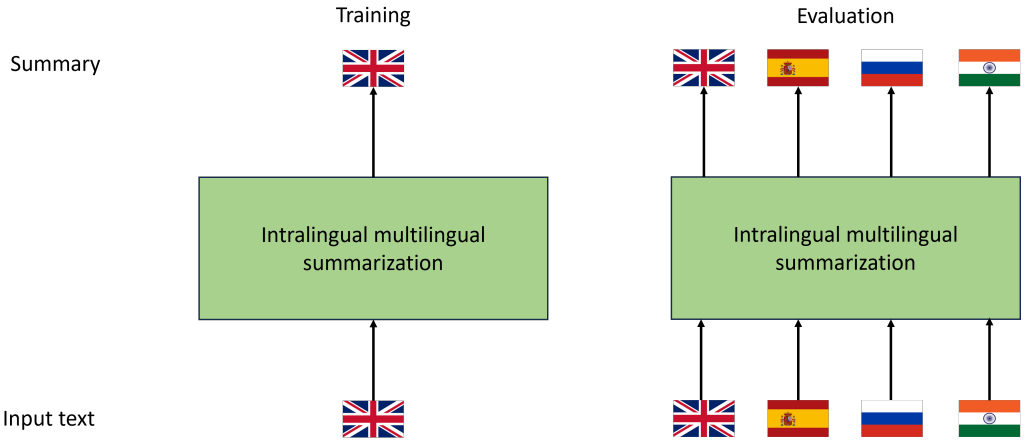

Figure 14: Intralingual multilingual summarization. Training is performed using only English data.

An advantage of the model trained with only English data is that it requires less supervised data. Many datasets for different tasks are available only in English. Being able to fine-tune a multilingual model using data from only one language brings us many benefits. This is especially relevant for low-resource languages, which often have much less supervised data. For this reason, I included both high-resource languages (English, Spanish, and Russian) and a low-resource language (Gujarati) in my experiments. I train models using only English data and evaluate zero-shot and few-shot scenarios for Spanish, Russian, and Gujarati.

A drawback of models trained only with English data is that they can become very English-specific. To overcome this problem, I also train models using multiple languages (English, Spanish, and Russian) and perform evaluations using only a low-resource language (Gujarati). Training with data in multiple languages is supposed to improve transfer learning between languages and hinder model weights from becoming

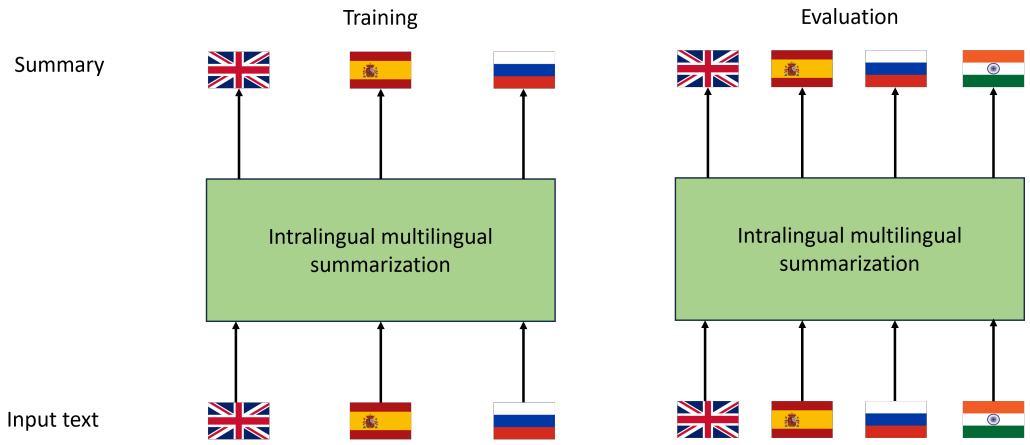

Figure 15: Intralingual multilingual summarization. Training is performed using English, Spanish, and Russian data.

language-specific, focusing more on understanding the input text and generating summaries. A disadvantage of this type of model is that it requires more supervised data. Ideally, this should be supervised data in different languages generated using the same scheme. For many tasks, there is a shortage of such data.

Through these intralingual experiments, it is possible to obtain answers to the first, third, and fourth research questions.

6.1 Dataset - XL-Sum

Hasan et al. (2021) introduced a new dataset for multilingual abstractive summarization called XL-Sum. This dataset contains over 1 million professionally annotated article-summary pairs extracted from the British Broadcasting Corporation (BBC) news website at `https://www.bbc.com/`. The dataset comprises data in 44 languages, including both high-resource and low-resource languages from various language families. A notable advantage of this dataset is that all the data in different languages originates from the same source. Consequently, all summaries are generated using the same summarization strategy across all languages, making this dataset a good choice for multilingual summarization that can be performed by a single model.

The XL-Sum dataset contains only monolingual data in various languages. Consequently, the languages of all abstractive summaries always correspond to the languages of the input articles. This dataset facilitates the development and evaluation of intralingual models capable of producing summaries in the same language as the articles. Table 7 provides an example in English illustrating how abstractive summaries are constructed, highlighting text segments that influence the generated summary.

Input Article: Yahoo's patents suggest users could weigh the type of ads against the sizes of discount before purchase. It says in two US patent applications that ads for digital book readers have been "less than optimal" to date. [...] "Greater levels of advertising, which may be more valuable to an advertiser and potentially more distracting to an e-book reader, may warrant higher discounts," it states. [...] It adds that the more willing the customer is to see the ads, the greater the potential discount. [...] At present, several Amazon and Kobo e-book readers offer full-screen adverts when the device is switched off and show smaller ads on their menu screens. [...] Yahoo does not currently provide ads to these devices, and a move into the area could boost its shrinking revenues.
Summary: Yahoo has signalled it is investigating e-book adverts as a way to stimulate its earnings.

Table 7: An example of the input article-summary pair in English (Hasan et al. 2021).

Table 8 displays the statistics of the XL-Sum dataset for four languages: English, Spanish, Russian, and Gujarati. It shows the number of samples available for the training, validation, and test splits.

language	train	validation	test
en	302627/459.92/22.31	11535/440.39/21.15	11535/437.29/21.24
es	35633/823.53/29.37	4763/766.47/27.42	4763/764.75/27.41
ru	60044/564.04/26.09	7780/466.26/24.21	7780/465.28/24.17
gu	8790/769.06/23.96	1139/542.60/21.23	1139/529.90/21.69

Table 8: Xlsum dataset statistics - number of samples/average input length in words/average output length in words.

6.2 Supervised oracle condition and baseline

In the beginning, I first establish the performance under oracle conditions with supervised data and calculate baseline results that can be used for further comparisons. The results of the supervised oracle model show values that I aim to achieve with different experiments.

Before performing supervised oracle and baseline experiments, it should be decided what training settings to use as a standard. Particularly, it is not clear whether it is better to update pre-trained token embeddings or not. I fine-tune two models using English, Spanish, and Russian data jointly and perform an evaluation using these three languages as supervised cases and additionally Gujarati as a zero-shot case.

It is observed that fine-tuning models without updating pre-trained token embeddings leads to better results in both supervised and zero-shot cases. Table 9 shows a comparison of two models: the first one with fine-tuned embeddings ("fine-tuned" scenario), and the second one with frozen embeddings ("frozen" scenario). The results of the model fine-tuned

without updating pre-trained token embeddings are better in all cases. This is especially notable in the zero-shot case evaluated with Gujarati data. Although the model generates texts as expected in Gujarati, the values of other metrics are much lower. It is remarkable that generated texts are much longer in this case (55.9 words against 14.4 words). This phenomenon can occur while using mBART models because of its design. The problem is that mBART models utilize a target language token as an end-of-sentence token. If models do not output this language token as expected, they continue generating texts, resulting in very long outputs with maximally allowed lengths. I suppose that in the case of token embeddings fine-tuning, models forget how to output target language tokens in other languages that are not available during fine-tuning. As the results of the first model are better in all cases, I perform all further experiments without fine-tuning embeddings.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
en	fine-tuned	39.3/16.4/31.4	78.3	100.0	17.1
	frozen	39.5/16.4/31.4	78.3	100.0	17.4
es	fine-tuned	33.9/12.9/25.8	74.0	100.0	19.3
	frozen	34.4/12.9/26.0	74.0	100.0	20.3
ru	fine-tuned	34.9/15.1/28.1	75.0	100.0	17.5
	frozen	35.3/15.3/28.4	75.0	100.0	17.3
gu (ZS)	fine-tuned	12.1/2.8/10.0	66.1	100.0	59.9
	frozen	17.8/5.2/16.0	71.8	99.6	14.4

Table 9: Models trained using English, Spanish, and Russian data jointly. 1) with embeddings fine-tuning ("fine-tuned") 2) without embeddings fine-tuning("frozen"). Evaluation is performed for English, Spanish, Russian, and Gujarati (zero-shot).

After the initial step, I conduct experiments that evaluate the performance under oracle conditions with supervised data and calculate baseline results. For a better comparison of further zero-shot and few-shot experiments, it is necessary to build three models initially. The first model ("supervised" scenario in Table 10) is fine-tuned under oracle conditions using supervised data in all four languages (English, Spanish, Russian, and Gujarati) jointly. The next two models are used for the calculation of baseline results that should be improved in further experiments. The second model ("ZS-baseline" scenario in Table 10) is fine-tuned using only English data. Using this model, I calculate zero-shot results for Spanish, Russian, and Gujarati data. The third model ("ZS-baseline (en+es+ru)" scenario in Table 10) is trained using data in several languages (English, Spanish, and Russian). Using this model, I calculate zero-shot results only for Gujarati. The results of these three models are provided in Table 10.

The results of the second model demonstrate that the performance of a model trained only with one language is very poor for zero-shot experiments in other languages. This trained model becomes very English-specific and leads to the off-target generation problem, generating summaries always in English, regardless of the input language. Additionally, it

generates very long summaries. Here the same problem occurs as described above with Gujarati data. In this case, the mBART model learns to use an English language token as an end-of-sequence token. During inference, it attempts to finish summaries by outputting the English language token. However, according to the design, it expects another language token and therefore continues with generation.

The results of the third model, evaluated with a zero-shot scenario in Gujarati, are much better. Training a model with several languages hinders it from becoming language-specific. It generates summaries in the expected language with good results but still worse than a supervised solution from the first model.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es	supervised	34.2/13.0/26.1	74.0	100.0	20.1
	ZS-baseline	7.9/1.2/6.2	66.0	0.2	61.7
ru	supervised	35.3/15.3/28.4	75.1	100.0	17.9
	ZS-baseline	1.2/0.3/1.1	64.3	2.3	59.5
gu	supervised	22.3/8.0/20.0	74.2	100.0	16.5
	ZS-baseline	1.6/0.3/1.5	59.2	13.4	61.3
	ZS-baseline (en+es+ru)	17.8/5.2/16.0	71.8	99.6	14.4

Table 10: Intralingual supervised learning and baseline results. Results of two baseline models: 1) using English data and evaluating using Spanish, Russian, and Gujarati 2) using English, Spanish, and Russian data jointly (en+es+ru) and evaluating using only Gujarati.

6.3 Translation-based results

I conduct experiments using translation models for both data pre- and post-processing. Since I work with intralingual data in different languages, it is necessary to employ translation models twice for each case.

Table 11 illustrates the results obtained using translation models and the summarization model trained with English data. Approaches utilizing both mBART50 and NLLB for translation perform similarly on Russian data. NLLB slightly outperforms mBART50 on Spanish data. However, a significant difference occurs when applying this method to Gujarati data. Although mBART50 generates text in Gujarati as expected, the results are notably poor. In contrast, NLLB performs significantly better with Gujarati data, for example, achieving a Rouge-1 value of 15.7 compared to 2.6. This suggests that NLLB works more effectively with low-resource languages.

When comparing these results with the zero-shot results from Table 10, it is notable that the translation-based solution performs much better than zero-shot cases applied to the model trained with only English data.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es	supervised	34.2/13.0/26.1	74.0	100.0	20.1
	mBART50	23.1/4.5/16.8	68.1	100.0	27.2
	NLLB	29.2/8.6/21.5	72.1	100.0	21.6
ru	supervised	35.3/15.3/28.4	75.1	100.0	17.9
	mBART50	26.5/8.4/21.3	72.4	99.8	15.2
	NLLB	26.7/8.5/21.3	72.4	100.0	15.3
gu	supervised	22.3/8.0/20.0	74.2	100.0	16.5
	mBART50	2.6/0.3/2.4	64.4	94.6	17.0
	NLLB	15.7/3.3/14.1	72.1	100.0	16.3

Table 11: Evaluation for Spanish, Russian, and Gujarati data using a model trained with only English data and translation models for translation to and from English.

6.4 Few-shot results

I perform few-shot experiments as described in Chapter 5.2. A model trained with English data is fine-tuned separately with Spanish, Russian, and Gujarati data. Additionally, a model trained with English, Spanish, and Russian data is fine-tuned using only Gujarati data.

Evaluations are conducted with different amounts of data: 10, 100, 1000, and 10000 (except for Gujarati, as there is less total data). Table 12 shows the results of few-shot experiments. Even with a small amount of added supervised data (10 samples), there is a noticeable improvement in performance. All summaries are generated in the expected languages. Increasing the amount of data brings the results closer to the supervised solution. For example, a model fine-tuned with 10 samples in Spanish results in a Rouge-1 value of 29.8 and an F-1 BertScore value of 71.8. Adding just 90 extra samples increases these values to 30.4 and 72.2, respectively. These results demonstrate that models can be easily adapted to other languages and do not require a large amount of data.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es	supervised	34.2/13.0/26.1	74.0	100.0	20.1
	FS 10	29.8/8.5/21.6	71.8	99.9	23.9
	FS 100	30.4/9.1/22.2	72.2	100.0	22.7
	FS 1000	31.9/10.4/23.6	72.9	100.0	21.5
	FS 10000	33.4/11.7/24.8	73.3	100.0	21.5
ru	supervised	35.3/15.3/28.4	75.1	100.0	17.9
	FS 10	29.6/10.2/23.2	73.0	100.0	17.1
	FS 100	30.6/10.9/23.6	73.2	100.0	19.5
	FS 1000	31.2/11.6/24.2	73.3	100.0	21.7
	FS 10000	31.7/12.3/24.5	73.4	100.0	23.4
gu	supervised	22.3/8.0/20.0	74.2	100.0	16.5
	FS 10	17.9/5.2/15.8	72.0	100.0	16.6
	FS 100	19.1/5.5/16.9	72.9	100.0	16.9
	FS 1000	20.9/6.7/18.5	73.5	100.0	17.0
	FS 10 (en+es+ru)	19.4/5.9/17.2	72.8	100.0	17.6
	FS 100 (en+es+ru)	20.6/6.4/18.2	73.5	100.0	18.0
	FS 1000 (en+es+ru)	21.5/6.9/19.1	73.8	100.0	16.8

Table 12: Few-shot results for Spanish, English, and Gujarati using a model trained with English data. And additionally few-shot results for Gujarati using a model trained with English, Spanish, and Russian data jointly (en+es+ru).

The few-shot experiments with Gujarati data also demonstrate that pre-training with multiple languages can be beneficial for further fine-tuning with low-resource languages. For example, fine-tuning a model with 10 samples, previously fine-tuned only with English data, results in a Rouge-1 value of 17.9. In contrast, fine-tuning the same model with 10 samples, previously fine-tuned with English, Spanish, and Russian data jointly, leads to a better Rouge-1 value of 19.4. This is also true for 100 and 1000 samples and all other metrics. Notably, fine-tuning a model with 1000 samples, previously fine-tuned with multiple languages, almost reaches values obtained using a supervised solution (Rouge-1 value of 21.5 compared to 22.3).

6.5 Zero-shot experiments with partial fine-tuning

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es	supervised	34.2/13.0/26.1	74.0	100.0	20.1
	ft. all	7.9/1.2/6.2	66.0	0.2	61.7
	ft. enc	26.2/7.3/19.4	70.8	85.2	23.6
	ft. SA, EA	29.7/8.6/21.8	71.9	99.5	20.4
	ft. Q and K	30.0/9.3/22.4	72.3	99.9	19.5
ru	supervised	35.3/15.3/28.4	75.1	100.0	17.9
	ft. all	1.2/0.3/1.1	64.3	2.3	59.5
	ft. enc	30.1/11.1/24.1	73.2	100.0	14.5
	ft. SA, EA	28.8/10.2/22.8	72.7	100.0	15.1
	ft. Q and K	30.8/11.8/24.7	73.6	100.0	14.8
gu	supervised	22.3/8.0/20.0	74.2	100.0	16.5
	ft. all	1.6/0.3/1.5	59.2	13.4	61.3
	ft. enc	17.2/5.2/15.8	71.7	100.0	12.3
	ft. SA, EA	12.4/3.4/11.1	68.6	99.8	17.0
	ft. Q and K	19.6/6.3/17.8	73.2	100.0	12.9
	ft. all (en+es+ru)	17.8/5.2/16.0	71.8	99.6	14.4
	ft. enc (en+es+ru)	18.2/5.4/16.6	72.2	100.0	12.7
	ft. SA, EA (en+es+ru)	19.1/6.1/17.2	72.6	100.0	13.3
	ft. Q and K (en+es+ru)	19.4/6.7/17.7	73.1	100.0	14.2

Table 13: Zero-shot results for Spanish, English, and Gujarati using models trained with English data. Additionally, zero-shot results for Gujarati using models trained with English, Spanish, and Russian data jointly ("en+es+ru"). "ft. all" updates all weights of all encoder and decoder layers; "ft. enc" updates all weights of all encoder layers; "ft. SA, EA" updates all weights of self-attention in encoder layers, encoder-attention in decoder layers, and normalization layers; "ft. Q and K" updates weights of queries and keys linear projections in self-attention of encoder layers and encoder-attention of decoder layers.

I conduct partial fine-tuning experiments using three different settings:

1. **ft. enc**: Update only the weights of encoder layers. This approach, proposed by Maurya et al. (2021) and Chi et al. (2020), is described in Section 3.1.1.
2. **ft. SA, EA**: Update only normalization layers and self-attention layers within the encoder layers. Within the decoder layers, update only normalization layers and encoder attention layers. This method, proposed by Li et al. (2021), is described in Section 3.1.1.

3. **ft. Q and K**: Update only the weights of keys and queries of self-attention in encoder layers and encoder attention in decoder layers. This approach, proposed in this thesis, is described in Section 4.1.

Table 13 illustrates the results of the experiments with partial fine-tuning. The conducted experiments confirm the results obtained by the authors who originally proposed these methods with partial fine-tuning. Both the first and the second settings lead to better zero-shot results and help generate summaries in the expected languages. However, determining which setting works better is challenging because results vary for different languages. For example, for Spanish, the second setting works better than the first method (Rouge-1 value of 29.7 compared to 26.2). But for data in Russian, it is the opposite (Rouge-1 value of 28.8 compared to 30.1).

An evaluation of the third setting proposed in this thesis shows that it performs even better than the first two. It results in some improvements for all conducted experiments. For example, for Russian, the Rouge-1 value is 30.8 which is higher than for the first (30.1) and the second (28.8) settings.

It is notable also in this case that pre-training performed using multiple languages helps in zero-shot cases with data in Gujarati. It gives an improvement of the BERTScore-F1 value for the first setting from 71.7 to 72.2 and for the second setting from 68.6 to 72.6. For the third setting, the results are very similar; some metrics are slightly better for the model pre-trained with English data (Rouge-1 value of 19.6 compared to 19.4), but some of them are better for the model pre-trained with English, Spanish, and Russian data jointly (Rouge-1 value of 6.7 compared to 6.3). It means that the third setting not only improves results in zero-shot cases but can also achieve such good results with less data.

Another advantage of partial fine-tuning is that it updates fewer model parameters and hence requires less time for fine-tuning. Table 14 demonstrates how many model parameters are trainable in different scenarios. The solution proposed in this thesis with fine-tuning only keys and queries updates fewer parameters than all other scenarios.

Scenario	# trainable parameters	% from overall
ft. all incl. embeddings	610,851,840	100%
ft. all excl. embeddings	352,718,848	57.7%
ft. enc	151,156,736	24.7%
ft. SA, EA	100,888,576	16.5%
ft. Q and K	50,380,800	8.2%

Table 14: Number of model parameters that are trainable during fine-tuning.

6.6 Results overview

This section provides a brief overview of the results of all approaches applied in this chapter for conducting experiments. Table 15 presents the Rouge-L and BERTScore-F1 values.

The best zero-shot approach, "ft. Q and K," outperforms the approach with translation models, "translated - NLLB," for all languages. Applying translation models for pre- and post-processing accumulates errors from the models and leads to slightly worse results. This is especially notable for Gujarati, where the Rouge-L value is 17.8 compared to 14.1. The reason for this could be that the performance of translation models for translation from and to low-resource languages is worse than for high-resource languages.

Comparing few-shot results, it is possible to say that about 1000 samples are required to achieve the best zero-shot approach results on average. Notably, Russian few-shot results with even 10000 samples perform worse than my proposed approach, achieving a BERTScore-F1 value of 73.4 compared to 73.6. Another observation is that few-shot experiments conducted for Gujarati perform better for the model pre-trained with multilingual data and require only 100 samples to obtain better results compared to the best zero-shot solution.

Scenario	es	ru	gu	gu (en+es+ru)
(1) supervised	26.1/74.0	28.4/75.1	20.0/74.2	20.0/74.2
(2) zero-shot (baseline)	6.2/66.0	1.1/64.3	1.5/59.2	16.0/71.8
(3) translated - NLLB	21.5/72.1	21.3/72.4	14.1/72.1	-
(4) (2) + 10 samples	21.6/71.8	23.2/73.0	15.8/72.0	17.2/72.8
(5) (2) + 100 samples	22.2/72.2	23.6/73.2	16.9/72.9	18.2/73.5
(6) (2) + 1000 samples	23.6/72.9	24.2/73.3	18.5/73.5	19.1/73.8
(7) (2) + 10000 samples	24.8/73.3	24.5/73.4	-	-
(8) ft. enc	19.4/70.8	24.1/73.2	15.8/71.7	16.6/72.2
(9) ft. SA, EA	21.8/71.9	22.8/72.7	11.1/68.6	17.2/72.6
(10) ft. Q and K	22.4/72.3	24.7/73.6	17.8/73.2	17.7/73.1

Table 15: Overview of Rouge-L and BERTScore-F1 values for all intralingual scenarios.

7 Crosslingual experiments

This section presents the results of cross-lingual experiments. Cross-lingual multilingual models can encode and generate texts in different languages, allowing for the input and output languages to differ. This allows to perform simultaneous summarization and translation. It is possible to construct more complicated scenarios using cross-lingual models.

In my thesis, I conduct experiments using only monolingual data in English, Spanish, and Russian during training. Applying different methods, I attempt to train models in such a way that cross-lingual summarization becomes possible. Figure 16 demonstrates such a model.

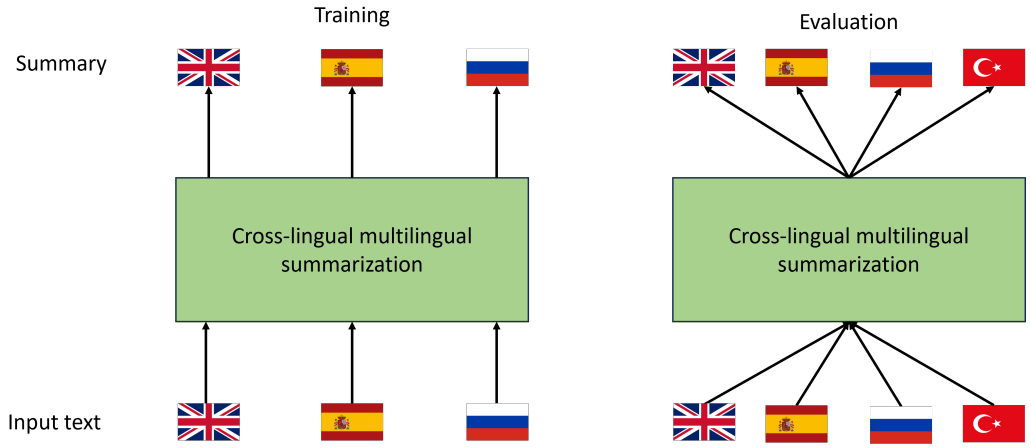

Figure 16: Cross-lingual multilingual summarization. Training is performed using monolingual English, Spanish, and Russian data jointly. Evaluation includes cross-lingual scenarios.

The performance of models is evaluated using cross-lingual data in different languages, considering combinations of languages from the second research question. The following language pairs are utilized:

- Spanish-English, Russian-English, and Spanish-Russian. All languages are available during the fine-tuning process but not in such combinations.
- Turkish-English. Only the output language is available during the fine-tuning process.
- English-Turkish. Only the input language is available during the fine-tuning process.

- Turkish-Turkish. Both input and output languages are not available during the fine-tuning process.

In this section, I will also conduct both zero-shot and few-shot experiments. With these experiments, I will be able to answer the second, third, and fourth research questions.

7.1 Dataset - WikiLingua

Ladhak et al. (2020) introduced another dataset for multilingual abstractive summarization known as WikiLingua. One specific feature and advantage of this dataset is that, in addition to monolingual data in various languages, it also includes cross-lingual data. This feature enables us to perform a broader range of experimental scenarios, making it even more attractive. The dataset comprises pairs of articles and abstractive summaries in 18 languages, all written by human authors.

Table 16 displays the statistics of the WikiLingua dataset for different language pairs used for experiments in this thesis: English-English, English-Turkish, Spanish-English, Spanish-Spanish, Spanish-Russian, Russian-English, Russian-Russian, Turkish-English, and Turkish-Turkish. It shows the number of samples available for the training, validation, and test splits.

languages	train	validation	test
en-en	95517/379.68/32.23	3000/379.82/32.64	27489/377.35/32.27
en-tr	3052/350.67/27.51	438/350.00/27.83	874/329.94/27.21
es-en	76295/406.96/31.74	3000/404.95/31.60	21726/406.76/32.31
es-es	76295/406.96/36.72	3000/404.95/36.62	21726/406.76/37.30
es-ru	32458/418.26/27.76	3000/428.64/28.35	8737/415.56/28.20
ru-en	35313/322.08/30.57	3000/326.38/31.75	9962/318.70/31.07
ru-ru	35313/322.08/27.79	3000/326.38/28.58	9962/318.70/28.12
tr-en	3052/265.21/32.65	438/263.42/32.60	874/249.35/32.33
tr-tr	3052/265.21/27.51	438/263.42/27.83	874/249.35/27.21

Table 16: WikiLingua dataset statistics - number of samples/average input length in words/average output length in words.

The data in the WikiLingua dataset is extracted from the WikiHow website (`https://www.wikihow.com/`). Similar to the XL-Sum dataset, articles and summaries in various languages within the WikiLingua dataset are constructed using the same scheme, making it a good choice for multilingual summarization tasks.

Articles in this dataset provide instructions with detailed steps explaining how to complete various tasks across a range of topics, such as *How to Make a Simple Chocolate Cake*, *How to Play Tennis*, and *How to Code*. Each task consists of multiple steps that offer comprehensive instructions on completing these tasks. Each step consists of a

textual paragraph with instructions and a one-sentence summary at the beginning of the paragraph, which serves as a summary of that step. All one-sentence summaries are combined into a single common summary, and all paragraphs are aggregated into an input article.

Table 17 presents an example of *How to ride a bicycle*, illustrating how task descriptions are transformed into articles and summaries in different languages.

Original task description: Step 1. Find a fitting location. When you're learning as a beginner, you want to find a place that's comfortable and far from traffic. A good place to start is a flat, smooth stretch of ground such as your driveway or your sidewalk. Those who don't have space at home can practice in a parking lot or park. Step 2. Wear riding clothing. Knee and elbow pads insulate joints and protect against scrapes, so they are recommended for all riders. Long-sleeved shirts and long pants also help protect against falls and can be combined with pads. Step 3. Put on a helmet. Helmets are recommended for beginners and experienced bike riders alike. You never know when an accident will happen. A broken bone can usually be fixed, but head trauma, common in bicycle accidents, leaves a lasting impact. Also, some areas have laws requiring riders to wear helmets.
Extracted input article: When you're learning as a beginner, you want to find a place that's comfortable and far from traffic. A good place to start is a flat, smooth stretch of ground such as your driveway or your sidewalk. Those who don't have space at home can practice in a parking lot or park. Knee and elbow pads insulate joints and protect against scrapes, so they are recommended for all riders. Long-sleeved shirts and long pants also help protect against falls and can be combined with pads. Helmets are recommended for beginners and experienced bike riders alike. You never know when an accident will happen. A broken bone can usually be fixed, but head trauma, common in bicycle accidents, leaves a lasting impact. Also, some areas have laws requiring riders to wear helmets.
Summary in English: Find a fitting location. Wear riding clothing. Put on a helmet.
Summary in German: Suche dir einen passenden Ort. Trage die richtige Kleidung. Setze einen Helm auf.

Table 17: A WikiLingua example from the article explaining how to ride a bicycle https://www.wikihow.com/Ride-a-Bicycle.

7.2 Supervised oracle condition and baseline

As in the case of intralingual experiments, initially, it is necessary to train models that will be utilized for further comparisons. For this purpose, two models are trained and evaluated:

1. The first model ("supervised" scenario in Table 18) is a model trained using supervised data that is used in other experiments for evaluation (Spanish-English, Russian-English, Spanish-Russian, Turkish-English, English-Turkish, Turkish-Turkish).
2. The second model ("baseline" scenario in Table 18) is a baseline model trained with monolingual data in English, Spanish, and Russian.

Table 18 demonstrates the results of the evaluation of the models described above. The baseline model performs very poorly for all cases, except for the monolingual language pair Turkish-Turkish. This result supports findings that were obtained with intralingual experiments that a model pre-trained with intralingual data in multiple languages performs relatively well in intralingual zero-shot cases.

For other language pairs, the model suffers from the off-target generation problem, making it inappropriate for zero-shot cross-lingual cases. It also generates very long summaries that are not expected. This problem and its explanation are described in Section 6.5.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es-en	supervised	38.4/14.8/31.4	78.1	99.8	24.3
	baseline	2.4/0.1/2.2	67.8	0.0	63.9
es-ru	supervised	21.0/6.2/18.0	75.2	100.0	19.6
	baseline	0.6/0.1/0.6	64.7	0.0	64.8
ru-en	supervised	36.0/13.2/29.4	77.5	99.7	23.6
	baseline	0.7/0.1/0.7	63.3	0.0	47.7
tr-en	supervised	41.5/18.2/34.5	78.8	99.4	25.2
	baseline	5.0/1.1/4.6	62.9	1.6	32.9
en-tr	supervised	23.8/8.1/20.7	73.2	100.0	20.2
	baseline	2.7/0.5/2.5	60.9	0.0	66.4
tr-tr	supervised	30.4/12.4/26.3	75.5	99.7	21.1
	baseline	20.9/6.0/18.0	71.5	96.4	20.9

Table 18: Cross-lingual supervised learning and baseline results.

7.3 Translation-based results

This section presents the results of cross-lingual summarization using translation models. For this purpose, it is necessary to train an intralingual summarization model using monolingual English data from the WikiLingua dataset. With the use of mBART50 and NLLB models, data can be pre- and postprocessed.

Table 19 demonstrates the results of the evaluation. Cross-lingual summarization using both translation models works well, especially for language pairs with English as an output language, where translation is performed only once to prepare the input. The NLLB model outperforms the mBART50 model on all language pairs, especially when Turkish is one of the languages in the language pair. These results align with those obtained during intralingual experiments in Section 6.3.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es-en	supervised	38.4/14.8/31.4	78.1	99.8	24.3
	mBART50	36.4/13.4/29.7	77.6	99.7	24.6
	NLLB	37.9/14.5/31.1	78.1	99.7	23.9
es-ru	supervised	21.0/6.2/18.0	75.2	100.0	19.6
	mBART50	16.1/3.5/13.9	73.3	100.0	20.0
	NLLB	16.7/3.7/14.4	73.8	100.0	19.5
ru-en	supervised	36.0/13.2/29.4	77.5	99.7	23.6
	mBART50	34.5/12.1/28.4	77.3	99.7	21.9
	NLLB	34.6/12.2/28.5	77.3	99.7	22.2
tr-en	supervised	41.5/18.2/34.5	78.8	99.4	25.2
	mBART50	36.1/13.0/29.5	77.1	100.0	26.3
	NLLB	40.9/17.0/34.1	78.7	99.4	24.8
en-tr	supervised	23.8/8.1/20.7	73.2	100.0	20.2
	mBART50	13.0/2.7/11.8	70.6	99.8	19.7
	NLLB	20.9/5.5/18.7	73.1	99.8	19.7
tr-tr	supervised	30.4/12.4/26.3	75.5	99.7	21.1
	mBART50	12.5/2.7/11.3	70.4	100.0	20.4
	NLLB	20.7/5.8/18.5	73.2	99.8	19.7

Table 19: Evaluation for different language pairs using a model trained with only English data and translation models for translation to and from English.

For language pairs generating summaries in English, the solution with the NLLB model is almost as good as the supervised solution. For instance, in the Turkish-English pair, the BERTScore-F1 value is 78.7, compared to 78.8 in the supervised scenario. The reason is that in this case, translation is performed only once as a pre-processing step from other languages to English. In other cases where summaries are generated in languages other than English, such as the Spanish-Russian pair, it is also necessary to translate the generated summaries from English to Russian, resulting in additional errors. Another

possible reason is that the NLLB model is likely more proficient at translating in English than in other languages.

7.4 Few-shot results

The results of few-shot experiments are presented in Table 20. The results show a similar performance as in intralingual cases in Section 6.4. For most language pairs, even a small amount of data (10 samples) significantly improves the results. However, the language pair English-Turkish requires more data, as 10 samples are insufficient for adaptation, and the model continues to generate texts in English. It means that it is more challenging for a model to adapt and generate summaries in a language unseen during the fine-tuning process. With a limited amount of data, it is almost possible to achieve a performance level comparable to that of a model trained with complete supervised data. For example, in the case of a Spanish-Russian language pair, a model fine-tuned with 10000 samples has a Rouge-1 value of 20.5 compared to 21.0 for the supervised solution.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es-en	supervised	38.4/14.8/31.4	78.1	99.8	24.3
	FS 10	33.6/10.5/26.9	76.3	99.3	27.5
	FS 100	34.3/11.0/27.4	76.7	99.7	27.0
	FS 1000	35.4/11.8/28.3	77.1	99.7	27.7
	FS 10000	37.4/13.6/30.2	77.8	99.7	26.8
es-ru	supervised	21.0/6.2/18.0	75.2	100.0	19.6
	FS 10	16.0/3.9/13.7	73.0	100.0	20.4
	FS 100	18.0/4.5/15.0	73.7	100.0	23.7
	FS 1000	19.1/5.0/16.0	74.1	100.0	23.4
	FS 10000	20.5/5.9/17.4	74.8	100.0	22.0
ru-en	supervised	36.0/13.2/29.4	77.5	99.7	23.6
	FS 10	31.0/9.5/25.2	76.0	99.7	24.0
	FS 100	32.7/10.2/26.2	76.3	99.8	26.2
	FS 1000	33.7/10.9/27.1	76.7	99.8	26.5
	FS 10000	35.3/12.4/28.8	77.3	99.7	24.7
tr-en	supervised	41.5/18.2/34.5	78.8	99.4	25.2
	FS 10	35.0/12.1/28.1	76.5	99.7	27.6
	FS 100	36.5/13.2/29.6	77.2	99.8	26.0
	FS 1000	39.1/15.5/32.1	78.2	99.6	25.9
en-tr	supervised	23.8/8.1/20.7	73.2	100.0	20.2
	FS 10	3.3/0.6/3.1	60.9	0.1	54.6
	FS 100	16.4/3.6/14.2	70.3	99.9	18.7
	FS 1000	21.3/5.9/18.4	72.4	99.9	19.6
tr-tr	supervised	30.4/12.4/26.3	75.5	99.7	21.1
	FS 10	25.7/8.5/21.9	74.0	99.7	21.6
	FS 100	27.7/9.5/23.5	74.9	99.8	22.6
	FS 1000	29.7/11.4/25.8	75.5	99.9	20.7

Table 20: Few-shot cross-lingual experiments results using a model trained with English data.

7.5 Zero-shot results with advanced methods

7.5.1 Adversarial language classifier and removing residual connections

I conduct experiments by applying an adversarial language classifier to the encoder output representation to prove that it is possible to train a cross-lingual model using only monolingual data. Results presented in Table 21 show that both approaches with different formulas ("(1) adv.loss" scenario stands for the original approach with Equation 3, "(2) adv.loss(KL-div)" scenario stands for the approach proposed in this thesis with Equation 5) work well for all language pairs that utilize an output language available during fine-tuning (English, Spanish, or Russian). For output languages that do not take part in the fine-tuning process (Turkish), the results are very poor. English-Turkish and Turkish-Turkish language pairs demonstrate bad results, as the fine-tuned model generates off-target summaries in unexpected languages in these cases. The Turkish-English language pair shows that this approach is also suitable for unseen input languages.

The proposed approach that utilizes Kullback–Leibler divergence slightly outperforms the original solution for all language pairs, particularly for pairs for which this solution with adversarial loss works. For example, in the Spanish-English pair, the Rouge-1 value is improved from 33.4 to 34.1, and the BERTScore-F1 value from 76.1 to 76.4. It means that it allows making an encoder output representation more language-independent.

I also conduct experiments with removing residual connections ("(2) + residual" scenario in Table 21) to eliminate position information from the encoder output representation. I apply this approach together with an adversarial loss proposed in this thesis with Equation 5. Since D. Liu, Niehues, et al. (2021) state that it is better to remove the residual connection in the middle layer and the encoder in the mBART model consists of 12 layers, I attempt to remove a residual connection in the seventh layer. Results demonstrate that this approach gives further improvements for cases when the input language is available during fine-tuning. In the same language pair Spanish-English it moves the results of the Rouge-1 and BERTScore-F1 values to 34.8 and 76.6. When the input language is not seen during fine-tuning, removing residual connections does not lead to any benefits.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es-en	supervised	38.4/14.8/31.4	78.1	99.8	24.3
	(1) adv.loss	33.4/10.5/26.7	76.1	98.5	28.6
	(2) adv.loss(KL-div)	34.1/11.0/27.2	76.4	99.4	28.9
	(2) + residual	34.8/11.3/27.6	76.6	99.7	29.9
es-ru	supervised	21.0/6.2/18.0	75.2	100.0	19.6
	(1) adv.loss	16.9/4.1/14.1	72.5	97.6	26.5
	(2) adv.loss(KL-div)	17.3/4.3/14.3	72.8	99.9	27.8
	(2) + residual	17.8/4.5/14.8	73.1	100.0	27.2
ru-en	supervised	36.0/13.2/29.4	77.5	99.7	23.6
	(1) adv.loss	31.9/9.9/25.3	75.7	99.6	29.2
	(2) adv.loss(KL-div)	32.2/10.2/25.6	75.8	99.8	29.3
	(2) + residual	33.1/10.6/26.3	76.1	99.9	29.8
tr-en	supervised	41.5/18.2/34.5	78.8	99.4	25.2
	(1) adv.loss	32.0/10.7/26.1	75.2	98.9	24.8
	(2) adv.loss(KL-div)	32.6/11.0/26.6	75.5	99.8	26.5
	(2) + residual	32.1/10.3/25.7	75.2	99.1	26.2
en-tr	supervised	23.8/8.1/20.7	73.2	100.0	20.2
	(1) adv.loss	2.8/0.5/2.5	60.9	0.0	65.4
	(2) adv.loss(KL-div)	2.8/0.5/2.6	60.9	0.0	65.4
	(2) + residual	2.7/0.5/2.5	61.0	0.0	64.9
tr-tr	supervised	30.4/12.4/26.3	75.5	99.7	21.1
	(1) adv.loss	5.7/1.1/5.2	62.8	10.6	55.1
	(2) adv.loss(KL-div)	3.5/0.5/3.2	61.1	0.1	63.4
	(2) + residual	2.5/0.4/2.3	60.8	0.0	59.2

Table 21: Results of applying adversarial language classifier and removing residual connections. "(1) adv.loss" - an original approach with Equation 3. "(2) adv.loss(KL-div)" - an approach proposed in this thesis with Equation 5. "(2) + residual" - a combination of the second approach with removing residual connections.

To evaluate how much language information tokens contain, it is helpful to train a language classifier on top of the encoder (Adi et al. 2017) that attempts to classify tokens based on their original languages. I perform this encoder output language classification for a baseline model and three models presented in this subsection. Figure 17 illustrates the results.

Conducted experiments demonstrate the successful implementation of all these approaches in making the encoder output more language-independent. The classification results correspond to the results discussed above and presented in Table 21. The original approach using Equation 3 works, but it leads to small changes and produces results that are significantly far from the uniform distribution. It reduces values for English, Spanish, and Russian from 0.992, 0.99, and 0.99 to 0.954, 0.953, and 0.956, respectively. The changes for Turkish are even smaller, decreasing from 0.985 to 0.968.

The approach proposed in this thesis, using Equation 5, demonstrates better performance and results and leads to more equal language probabilities. English, Spanish, and Russian become less predictable, with probabilities of 0.594, 0.689, and 0.641, respectively. The probability for Turkish also decreases, although remaining relatively high at 0.905, but lower than the original value. The problem is that for languages participating in fine-tuning with the adversarial loss it is attempted to eliminate their specific language features, but some of the specific features of other languages (in this case Turkish) remain unchanged. Conducting such fine-tuning with more languages, including those from the same language family, could potentially lead to better results, even for languages not seen during the fine-tuning process.

Combining the proposed approach with removing residual connections further improves the results, bringing them closer to a language-independent representation with probabilities of 0.496, 0.523, 0.51, and 0.644 for English, Spanish, Russian, and Turkish, respectively. Notably, English and Spanish have more similar probabilities with each other than with Russian, which could be explained by linguistic similarities between these languages. They share the same alphabet and have common tokens.

Another interesting observation is that it becomes more difficult to classify Turkish, although it does not participate in the original fine-tuning. There could be a combination of two reasons for that 1) the effectiveness of this approach, as it removes direct connections from input tokens, and 2) the potential loss of some language-specific features learned during pre-training with extensive data, as I modify the architecture of one encoder layer in this approach, requiring the model to relearn these features.

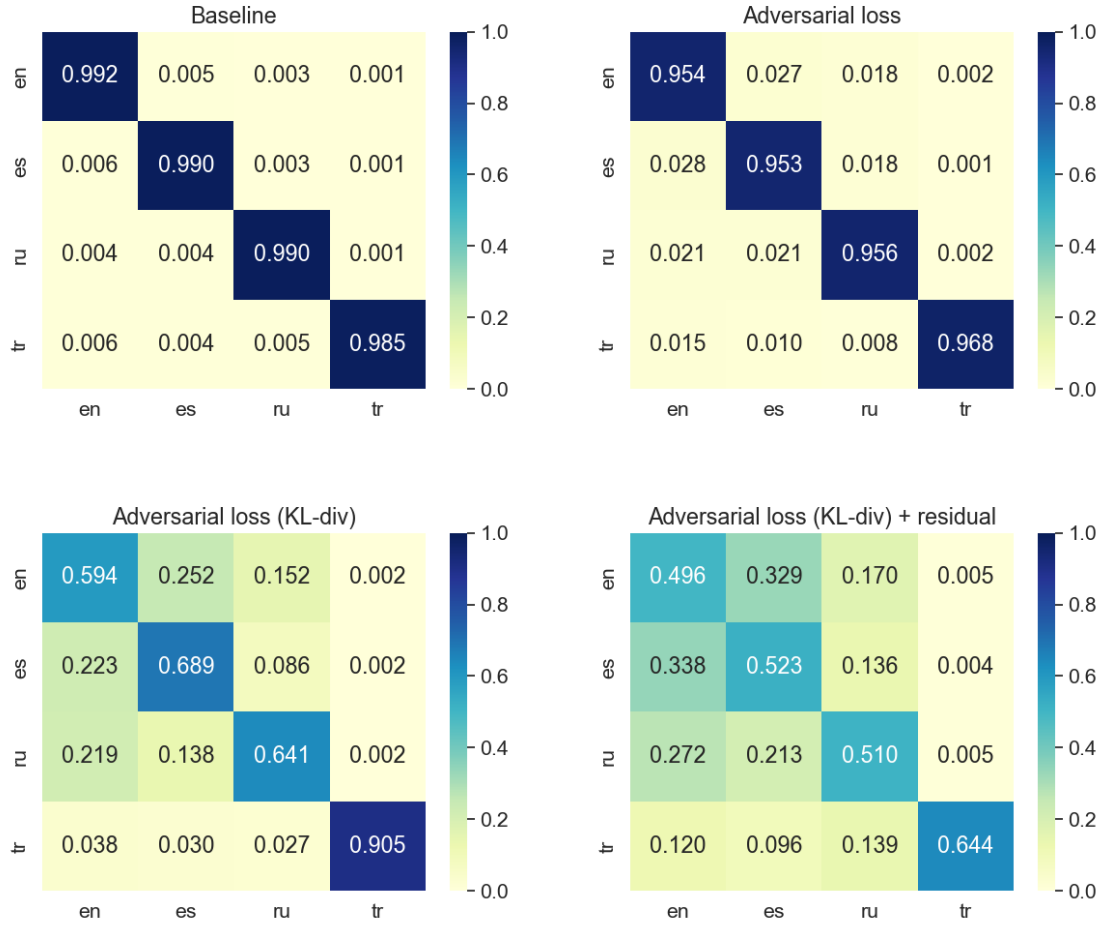

Figure 17: Encoder output language classification results. "Baseline" - a baseline model trained without any advanced approaches. "Adversarial loss" - an original approach with Equation 3. "Adversarial loss (KL-div)" - an approach proposed in this thesis with Equation 5. "Adversarial loss (KL-div) + residual" - a combination of the second approach with removing residual connections.

7.5.2 Language adapters

I conduct experiments with two types of adapters as described in Section 4.3: layers adapters and encoder output adapters. Table 22 demonstrates the results of experiments.

A solution utilizing adapters proposed by Philip et al. (2020) ("layer" scenario) gives improvements for some language pairs (Russian-English and Turkish-English) in comparison to zero-shot cases. At the same time, the results for other pairs (Spanish-English and Spanish-Russian) are poor.

Language adapters that I proposed to use in this thesis ("encoder output" scenario) and applied at the token level to encoder output representation show good results for all language pairs. They enable good zero-shot performance, generating summaries in the correct languages with expected content.

Encoder output adapters outperform layer adapters significantly. The problem with layer adapters is that they contain residual connections (see 6). Even with the application of language-specific adapters, the original signal coming from the input language stays strong and has an influence on the output language. In contrast, encoder output adapters, do not contain a residual connection and just perform language-specific linear projections. This enables them to change values much stronger and to adapt encoder output tokens to the expected output language.

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es-en	supervised	38.4/14.8/31.4	78.1	99.8	24.3
	layer	4.5/0.5/4.2	70.6	1.2	25.4
	encoder output	32.5/10.0/26.2	76.3	99.6	24.4
es-ru	supervised	21.0/6.2/18.0	75.2	100.0	19.6
	layer	0.7/0.1/0.7	64.9	0.3	56.8
	encoder output	16.6/4.0/13.9	73.1	100.0	23.1
ru-en	supervised	36.0/13.2/29.4	77.5	99.7	23.6
	layer	20.9/5.8/17.1	70.8	57.9	19.4
	encoder output	30.6/9.3/24.7	75.8	99.7	23.2
tr-en	supervised	41.5/18.2/34.5	78.8	99.4	25.2
	layer	16.7/4.8/14.1	68.4	49.4	17.2
	encoder output	31.9/10.4/25.9	75.7	99.8	22.0

Table 22: Results of experiments with language adapters. "Layer" - language adapters applied to layers and proposed by Philip et al. (2020). "encoder output" - language adapters applied to encoder output representation and proposed in this thesis.

7.5.3 Two-step fine-tuning (translate-then-summarize)

Another possible strategy to train a multilingual summarization model capable of performing cross-lingual summarization is training in two steps. With two-step fine-tuning, it is possible to first fine-tune a model with one task and after that fine-tune it again with another task. This training approach utilizes transfer learning between tasks and improves results.

In the context of cross-lingual summarization, two-step fine-tuning consists of the following steps:

1. **Fine-tune with a translation task:** In this step, a model learns to perform translation between desired languages. This fine-tuning step allows the model to learn language-specific features and patterns relevant to translation. I perform translation using many-to-many data in all four languages that I use for evaluation: English, Spanish, Russian, and Turkish.

   To prepare the translation data for this step, I parse a WikiLingua dataset presented at the beginning of this chapter. I iterate over samples in different language pairs and matched samples that have the same input text or output summary in the same language, but the corresponding output summary or input text is presented in different languages. By performing such matching, I generate translation data in the same domain as used for summarization. A translation model trained with such data is capable of translating both short and long sequences.

2. **Fine-tune with a summarization task:** In this step, a model learns how to summarize texts, adapting to the characteristics of summarization. To avoid the problem of catastrophic forgetting, the same technique with partial key-query fine-tuning as described in Chapter 4.1 is applied. Similar to intralingual cases, I apply key-query fine-tuning to both the encoder and decoder. I fine-tune two models: the first one using only English data, and the second one using English, Spanish, and Russian data jointly.

An advantage of this approach with two-step fine-tuning is that there is much more cross-lingual supervised data for a translation task than for other NLP tasks, such as summarization. This advantage is especially important for language combinations, including low-resource languages, as fine-tuning with a summarization task does not require supervised data in all language pairs.

Table 23 presents the results of the two-step approach. All cases confirm the effectiveness of this approach, showing comparable results with the translation-based solution using the NLLB model. Notably, the generated summaries in the two-step approach are always longer, resulting in lower BERTScore-F1 values, but, in some cases, higher Rouge-1 values. For instance, in the Spanish-Russian pair, the generated summaries have an average length of 24.3 for a model fine-tuned using only English data and 22.3 for a model fine-tuned using English, Spanish, and Russian data jointly. In comparison, the translation-based solution has an average length of 19.5 words which is a little bit less. The BERTScore-F1 value is 73.8 in the translation-based approach, compared to 73.6 and 73.4 for the two-step

Language pair	Scenario	Rouge-1/2/L	BERTScore-F1	Expected language	Length
es-en	supervised	38.4/14.8/31.4	78.1	99.8	24.3
	NLLB	37.9/14.5/31.1	78.1	99.7	23.9
	en	35.4/12.2/28.4	76.9	99.6	26.5
	en+es+ru	34.4/11.5/27.7	76.5	98.4	26.1
es-ru	supervised	21.0/6.2/18.0	75.2	100.0	19.6
	NLLB	16.7/3.7/14.4	73.8	100.0	19.5
	en	17.9/4.6/14.8	73.6	99.5	24.3
	en+es+ru	17.6/4.6/14.8	73.4	98.4	22.3
ru-en	supervised	36.0/13.2/29.4	77.5	99.7	23.6
	NLLB	34.6/12.2/28.5	77.3	99.7	22.2
	en	33.7/11.0/26.7	76.2	99.8	28.9
	en+es+ru	32.8/10.7/26.3	76.1	99.6	25.3
tr-en	supervised	41.5/18.2/34.5	78.8	99.4	25.2
	NLLB	40.9/17.0/34.1	78.7	99.4	24.8
	en	39.1/15.1/31.1	77.4	99.8	30.4
	en+es+ru	38.6/14.6/30.7	77.4	99.7	29.6
en-tr	supervised	23.8/8.1/20.7	73.2	100.0	20.2
	NLLB	20.9/5.5/18.7	73.1	99.8	19.7
	en	14.7/4.1/12.5	68.8	66.6	27.9
	en+es+ru	20.4/5.8/16.7	71.3	98.7	28.4
tr-tr	supervised	30.4/12.4/26.3	75.5	99.7	21.1
	NLLB	20.7/5.8/18.5	73.2	99.8	19.7
	en	20.4/6.1/16.3	70.7	92.4	36.6
	en+es+ru	23.1/7.0/18.4	72.0	100.0	37.3

Table 23: Two-step fine-tuning (translate-then-summarize) results. "en" - only English data was used for fine-tuning in the summarization step. "en+es+ru" - English, Spanish, and Russian data was used for fine-tuning in the summarization step.

approach. However, the Rouge-1 value is only 16.7, as opposed to 17.9 and 17.6 in the two-step approach.

I also apply the second fine-tuning step with a summarization task to the mBART50 (pre-trained for many-to-many translation version) and NLLB models. This approach does not work for both models. The problem with mBART50 is that it is pre-trained using many-to-one and one-to-many data using English-centric data. This model can not perform intralingual transformations, resulting in the off-target generation. This model can not learn how to summarize texts by updating only queries and keys while using only monolingual data. It leads to significant losses but at the same time, it can not update enough weights because of partial fine-tuning. The problem with the NLLB model arises from its limitation to inputs of a maximum of 512 tokens. Many WikiLingua input texts are longer than 512 tokens. It makes it difficult for the model to establish relationships between truncated shorter input texts and expected summaries.

7.6 Results overview

This section provides a brief overview of the results of all approaches applied in this chapter for conducting experiments. Table 24 presents the Rouge-L and BERTScore-F1 values.

All proposed advanced approaches applying language-specific adapters and an adversarial language classifier loss work well in zero-shot cases. However, they are not capable of generating summaries in languages unseen during fine-tuning, e.g., English-Turkish. A good fact is that they can generate summaries from input texts in languages unseen during fine-tuning, e.g., Turkish-English. To conduct summaries in languages unseen during fine-tuning, the translation-based solution, the two-step solution, or few-shot scenarios can be used.

The best zero-shot solution, "(9) + residual," outperforms the approach with translation models only for the Spanish-Russian pair (the Rouge-L value is 14.8 compared to 14.4). For other pairs, where English is the output language, the translation-based approach works better. As described in Section 7.3, possible reasons for this are that in the Spanish-Russian case, translation is conducted twice for both pre- and post-processing data, resulting in additional errors, and that the NLLB model is likely more proficient at translating in English than in other languages.

Scenario	es-en	es-ru	ru-en	tr-en	en-tr	tr-tr
(1) supervised	31.4/78.1	18.0/75.2	29.4/77.5	34.5/78.8	20.7/73.2	26.3/75.5
(2) zero-shot (baseline)	2.2/67.8	0.6/64.6	0.7/63.3	4.6/62.9	2.5/60.9	18.0/71.5
(3) translated - NLLB	31.1/78.1	14.4/73.8	28.5/77.3	34.1/78.7	18.7/73.1	18.5/73.2
(4) (2) + 10 samples	26.9/76.3	13.7/73.0	25.2/76.0	28.1/76.5	3.1/60.9	21.9/74.0
(5) (2) + 100 samples	27.4/76.7	15.0/73.7	26.2/76.3	29.6/77.2	14.2/70.3	23.5/74.9
(6) (2) + 1000 samples	28.3/77.1	16.0/74.1	27.1/76.7	32.1/78.2	18.4/72.4	25.8/75.5
(7) (2) + 10000 samples	30.2/77.8	17.4/74.8	28.8/77.3	-	-	-
(8) adv.loss	26.7/76.1	14.1/72.5	25.3/75.7	26.1/75.2	2.5/60.9	5.2/62.8
(9) adv.loss(KL-div)	27.2/76.4	14.3/72.8	25.6/75.8	26.6/75.5	2.6/60.9	3.2/61.1
(10) (9) + residual	27.6/76.6	14.8/73.1	26.3/76.1	25.7/75.2	2.5/61.0	2.3/60.8
(11) Dec. layer adapters	4.2/70.6	0.7/64.9	17.1/70.8	14.1/68.4	-	-
(12) Enc. output adapters	26.2/76.3	13.9/73.1	24.7/75.8	25.9/75.7	-	-
(13) Two-step - en	28.4/76.9	14.8/73.6	26.7/76.2	31.1/77.4	12.5/68.8	16.3/70.7
(14) Two-step - en+es+ru	27.7/76.5	14.8/73.4	26.3/76.1	30.7/77.4	16.7/71.3	18.4/72.0

Table 24: Overview of Rouge-L and BERTScore-F1 values for all cross-lingual scenarios.

To achieve better results with few-shot scenarios than with the best zero-shot solution, "(9) + residual," about 100-1000 samples are required. For input languages that do not take part in fine-tuning, e.g., Turkish, few-shot results with 10 samples are already better than advanced approaches. The value of the Rouge-L score in the "(2) + 10 samples" case is 28.1, compared to 26.6 in the "adv.loss(KL-div)" case. The reason for that is that advanced approaches applied in this thesis also change the weights of the encoder, making it slightly

more language-specific and adapting it more to languages that take part in fine-tuning. In this case, the transfer learning between languages seen and unseen during fine-tuning becomes smaller.

8 Conclusion

This chapter concludes this thesis, in which I analyzed different approaches that can be applied to improve multilingual abstractive summarization. I conducted zero-shot and few-shot experiments using the pre-trained mBART model. It was proven that it is possible to obtain very good results with the use of few or even no data for both intralingual and cross-lingual cases. Section 8.1 presents the answers to research questions stated in Chapter 1. Section 8.2 discusses potential extensions to this thesis in the future.

8.1 Answers to research questions

Research question 1: How can we generate intralingual summaries of texts in low-resource languages when we only have intralingual data available in other languages and a pre-trained multilingual model?

In Chapter 6, I conducted different experiments with intralingual data. Fine-tuning a model using data only in one language without applying advanced approaches makes the model language-specific. Zero-shot experiments showed that this approach leads to the off-target generation problem. Fine-tuning a model using data in multiple languages can help avoid this problem. A model fine-tuned with multiple languages performs much better in zero-shot cases, enabling the summarization of texts in low-resource languages.

Research question 2: How can we generate cross-lingual summaries of texts when we only have intralingual data available in the same languages or other languages, along with a pre-trained multilingual model?

Chapter 7 describes the results of cross-lingual experiments. A model fine-tuned using only intralingual data in multiple languages without applying advanced approaches can not generate cross-lingual summaries. Such a model remains intralingual and suffers from the off-target problem as well. A language token used by pre-trained multilingual models, e.g., mBART, is not enough to force the output to be in the expected language. It was shown that the encoder output representation remains very language-dependent in this case and the output language depends much more on it than on the language token.

Research question 3: What is the required amount of supervised data for few-shot experiments to achieve results that are comparable to experiments with complete supervised data in both intralingual and cross-lingual scenarios?

Experiments conducted for both intralingual and cross-lingual cases showed that even a small amount of supervised data in the desired languages (10 samples) can significantly

improve the results. Adding more data further improves the results. In most scenarios, fine-tuning with 1000 or 10000 samples leads to results close to the supervised solution.

Research question 4: What methods can be applied to improve zero-shot results in both intralingual and cross-lingual scenarios?

I evaluated different approaches from the literature and new approaches proposed in this thesis. It was proven that partial fine-tuning helps against off-target generation and maximizes benefits from the use of pre-trained multilingual models. This method improves the results of zero-shot experiments with intralingual models significantly. The newly proposed approach, fine-tuning only queries and keys of multi-head attention, showed the best results, resulting in better metric values and requiring less time for fine-tuning as it updates fewer parameters.

Applying language-specific encoder output adapters proposed in this thesis makes it possible to perform cross-lingual summarization in zero-shot cases. Another suitable approach for that is using an adversarial loss that was proposed earlier for machine translation. This method works well for cross-lingual summarization as it makes the encoder output representation less language-specific. The improvement proposed in this thesis that better encourages language-independent encoder output representation by using Kullback–Leibler divergence to a uniform class distribution as a loss function leads to even better results. This new method was also combined with another approach that removes a residual connection from one encoder layer to decrease positional information from the encoder output. This combination showed the best results. I also performed a probing analysis of the encoder output representation trying to recognize the input language. This analysis showed that new approaches also lead to more language-independent results.

8.2 Future work

It should be further analyzed how to make the encoder output representation less language-specific, both at the token and sequence levels. Even by making tokens less language-specific, it is still possible to identify the input language. Different languages have distinct grammar rules; the same words in different languages can be tokenized into various numbers of tokens, influencing the length of sequences; and some words simply do not exist in certain languages. All these challenges make it difficult to achieve a more language-independent encoder output.

For many cross-lingual sequence-to-sequence tasks, it would be beneficial to pre-train a cross-lingual version of mBART that utilizes multiple languages and enables high-quality transformations. Currently available pre-trained mBART versions are either intralingual or cross-lingual, showing poor results for many language directions, especially with low-resource languages. Some currently available cross-lingual models, such as NLLB, can not accept inputs longer than 512 tokens, making them not applicable for some tasks. Pre-training the cross-lingual version of mBART could be conducted using the advanced

approaches proposed in this thesis. Applying these approaches during pre-training could lead to even better results because of the amount of available unsupervised data. Such pre-training would cover more tokens during training and would not suffer from overfitting. However, such pre-training would require significant computation resources, and therefore, it was out of the scope of this thesis.

Bibliography

Adi, Yossi et al. (2017). *Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks*. arXiv: 1608.04207 [cs.CL].

Allahyari, Mehdi and Saeid Safaei (2017). "Summarization Techniques : A Brief Survey". In: URL: https://api.semanticscholar.org/CorpusID:7620732.

Arivazhagan, Naveen et al. (2019). *The Missing Ingredient in Zero-Shot Neural Machine Translation*. arXiv: 1903.07091 [cs.CL].

Bapna, Ankur and Orhan Firat (Nov. 2019). "Simple, Scalable Adaptation for Neural Machine Translation". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, pp. 1538–1548. DOI: 10.18653/v1/D19-1165. URL: https://aclanthology.org/D19-1165.

Chi, Zewen et al. (Apr. 2020). "Cross-Lingual Natural Language Generation via Pre-Training". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34, pp. 7570–7577. DOI: 10.1609/aaai.v34i05.6256.

Chopra, Sumit, Michael Auli, and Alexander M. Rush (June 2016). "Abstractive Sentence Summarization with Attentive Recurrent Neural Networks". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, pp. 93–98. DOI: 10.18653/v1/N16-1012. URL: https://aclanthology.org/N16-1012.

Chung, Junyoung et al. (2014). "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *CoRR* abs/1412.3555. arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555.

Devlin, Jacob et al. (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

French, Robert M. (1999). "Catastrophic forgetting in connectionist networks". In: *Trends in Cognitive Sciences* 3.4, pp. 128–135. ISSN: 1364-6613. DOI: https://doi.org/10.1016/S1364-6613(99)01294-2. URL: https://www.sciencedirect.com/science/article/pii/S1364661399012942.

Gialitsis, Nikolaos, Nikiforos Pittaras, and Panagiotis Stamatopoulos (Sept. 2019). "A topic-based sentence representation for extractive text summarization". In: *Proceedings of the Workshop MultiLing 2019: Summarization Across Languages, Genres and Sources*. Varna,

Bulgaria: INCOMA Ltd., pp. 26–34. DOI: 10.26615/978-954-452-058-8_005. URL: https://aclanthology.org/W19-8905.

Hasan, Tahmid et al. (Aug. 2021). "XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Ed. by Chengqing Zong et al. Online: Association for Computational Linguistics, pp. 4693–4703. DOI: 10.18653/v1/2021.findings-acl.413. URL: https://aclanthology.org/2021.findings-acl.413.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long Short-Term Memory". In: *Neural Comput.* 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

Kudo, Taku and John Richardson (Nov. 2018). "SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Ed. by Eduardo Blanco and Wei Lu. Brussels, Belgium: Association for Computational Linguistics, pp. 66–71. DOI: 10.18653/v1/D18-2012. URL: https://aclanthology.org/D18-2012.

Ladhak, Faisal et al. (Nov. 2020). "WikiLingua: A New Benchmark Dataset for Cross-Lingual Abstractive Summarization". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Ed. by Trevor Cohn, Yulan He, and Yang Liu. Online: Association for Computational Linguistics, pp. 4034–4048. DOI: 10.18653/v1/2020.findings-emnlp.360. URL: https://aclanthology.org/2020.findings-emnlp.360.

Lewis, Mike et al. (July 2020). "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://aclanthology.org/2020.acl-main.703.

Li, Xian et al. (Aug. 2021). "Multilingual Speech Translation from Efficient Finetuning of Pretrained Models". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Ed. by Chengqing Zong et al. Online: Association for Computational Linguistics, pp. 827–838. DOI: 10.18653/v1/2021.acl-long.68. URL: https://aclanthology.org/2021.acl-long.68.

Lin, Chin-Yew (July 2004). "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, pp. 74–81. URL: https://aclanthology.org/W04-1013.

Liu, Danni and Jan Niehues (Dec. 2022). "Learning an Artificial Language for Knowledge-Sharing in Multilingual Translation". In: *Proceedings of the Seventh Conference on Machine Translation (WMT)*. Ed. by Philipp Koehn et al. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, pp. 188–202. URL: https://aclanthology.org/2022.wmt-1.12.

Liu, Danni, Jan Niehues, et al. (Aug. 2021). "Improving Zero-Shot Translation by Disentangling Positional Information". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Ed. by Chengqing Zong et al.

Online: Association for Computational Linguistics, pp. 1259–1273. DOI: 10.18653/v1/2021.acl-long.101. URL: https://aclanthology.org/2021.acl-long.101.

Liu, Yang and Mirella Lapata (Nov. 2019). "Text Summarization with Pretrained Encoders". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, pp. 3730–3740. DOI: 10.18653/v1/D19-1387. URL: https://aclanthology.org/D19-1387.

Liu, Yinhan et al. (2020). "Multilingual Denoising Pre-training for Neural Machine Translation". In: *Transactions of the Association for Computational Linguistics* 8. Ed. by Mark Johnson, Brian Roark, and Ani Nenkova, pp. 726–742. DOI: 10.1162/tacl_a_00343. URL: https://aclanthology.org/2020.tacl-1.47.

Lui, Marco and Timothy Baldwin (July 2012). "langid.py: An Off-the-shelf Language Identification Tool". In: *Proceedings of the ACL 2012 System Demonstrations*. Jeju Island, Korea: Association for Computational Linguistics, pp. 25–30. URL: https://aclanthology.org/P12-3005.

Maurya, Kaushal Kumar et al. (Aug. 2021). "ZmBART: An Unsupervised Cross-lingual Transfer Framework for Language Generation". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Ed. by Chengqing Zong et al. Online: Association for Computational Linguistics, pp. 2804–2818. DOI: 10.18653/v1/2021.findings-acl.248. URL: https://aclanthology.org/2021.findings-acl.248.

Mikolov, Tomás et al. (2013). "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1301.3781.

Napoles, Courtney, Matthew Gormley, and Benjamin Van Durme (June 2012). "Annotated Gigaword". In: *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*. Montréal, Canada: Association for Computational Linguistics, pp. 95–100. URL: https://aclanthology.org/W12-3018.

Ott, Myle et al. (June 2019). "fairseq: A Fast, Extensible Toolkit for Sequence Modeling". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Ed. by Waleed Ammar, Annie Louis, and Nasrin Mostafazadeh. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 48–53. DOI: 10.18653/v1/N19-4009. URL: https://aclanthology.org/N19-4009.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162.

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.

Pilault, Jonathan et al. (Nov. 2020). "On Extractive and Abstractive Neural Document Summarization with Transformer Language Models". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Bonnie Webber et al. Online: Association for Computational Linguistics, pp. 9308–9319. DOI: 10.18653/v1/2020.emnlp-main.748. URL: https://aclanthology.org/2020.emnlp-main.748.

Radford, Alec et al. (2019). "Language Models are Unsupervised Multitask Learners". In: URL: https://api.semanticscholar.org/CorpusID:160025533.

Raffel, Colin et al. (2020). "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *Journal of Machine Learning Research* 21.140, pp. 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

Ratcliff, Roger (1990). "Connectionist models of recognition memory: constraints imposed by learning and forgetting functions." In: *Psychological review* 97 2, pp. 285–308. URL: https://pubmed.ncbi.nlm.nih.gov/2186426/.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). "Learning internal representations by error propagation". In: URL: https://api.semanticscholar.org/CorpusID:62245742.

Team, NLLB et al. (2022). *No Language Left Behind: Scaling Human-Centered Machine Translation*. arXiv: 2207.04672 [cs.CL].

Touvron, Hugo et al. (2023). *LLaMA: Open and Efficient Foundation Language Models*. arXiv: 2302.13971 [cs.CL].

Vaswani, Ashish et al. (2017). "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., pp. 6000–6010. ISBN: 9781510860964.

Vu, Tu et al. (Dec. 2022). "Overcoming Catastrophic Forgetting in Zero-Shot Cross-Lingual Generation". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, pp. 9279–9300. DOI: 10.18653/v1/2022.emnlp-main.630. URL: https://aclanthology.org/2022.emnlp-main.630.

Wang, Thomas et al. (17–23 Jul 2022). "What Language Model Architecture and Pretraining Objective Works Best for Zero-Shot Generalization?" In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, pp. 22964–22984. URL: https://proceedings.mlr.press/v162/wang22u.html.

Wenzek, Guillaume et al. (May 2020). "CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data". English. In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*. Ed. by Nicoletta Calzolari et al. Marseille, France: European Language Resources Association, pp. 4003–4012. ISBN: 979-10-95546-34-4. URL: https://aclanthology.org/2020.lrec-1.494.

Xue, Linting et al. (June 2021). "mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer". In: *Proceedings of the 2021 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies. Ed. by Kristina Toutanova et al. Online: Association for Computational Linguistics, pp. 483–498. DOI: 10.18653/v1/2021.naacl-main.41. URL: https://aclanthology.org/2021.naacl-main.41.

Yang, Zhen et al. (July 2018). "Unsupervised Neural Machine Translation with Weight Sharing". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 46–55. DOI: 10.18653/v1/P18-1005. URL: https://www.aclweb.org/anthology/P18-1005.

Zhang, Biao et al. (July 2020). "Improving Massively Multilingual Neural Machine Translation and Zero-Shot Translation". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, pp. 1628–1639. DOI: 10.18653/v1/2020.acl-main.148. URL: https://aclanthology.org/2020.acl-main.148.

Zhang*, Tianyi et al. (2020). "BERTScore: Evaluating Text Generation with BERT". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SkeHuCVFDr.