University of Helsinki
Faculty of Arts
Department of Language Technology

Master's Thesis

Exploring model robustness for Language Normalization tasks when using Knowledge Distillation for handling previously unseen Code-Switched data

Daniel Pietschke

Master's Programme in Linguistic Diversity and Digital Humanities

Supervisor: Dr. Jörg Tiedemann 30.11.2025

Acknowledgments

In writing this thesis, while my own effort, I would not have been able to complete it all by myself without the support of a number of people. Professionally, I would love to thank Ona De Gibert Bonet for always helping me out to the best of her abilities whenever the OpusDistillery pipeline had an issue I could not solve myself, and also patiently explaining the entire process as well as walking me through setting it up. Furthermore, I would like to thank Christina Mercado for providing the translations of the sentences for the sentence analysis as a native speaker of Mexican Spanish and American English. Without her insights into the language and its nuances, as well as colloquial terms, a lot of important information about the sentence analysis would not have been possible.

Finally, special thanks go to my girlfriend, Veroonika Remets, for always believing in me, pushing me to achieve the best results in my work, and always having an open ear whenever I felt lost and overwhelmed. This thesis would not exist in this form without her, possibly not at all.

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Tiedekunta – Fakultet – Faculty	Koulutusohjelma – Utbildningsprogram – Degree Programme	
Opintosuunta – Studieinriktning – Study Track		
Tekijä – Författare – Author		
Työn nimi – Arbetets titel – Title		
Työn laji – Arbetets art – Level	Aika – Datum – Month and year	Sivumäärä– Sidoantal – Number of pages
Tiivistelmä – Referat – Abstract		
Avainsanat – Nyckelord – Keywords		
Säilytyspaikka – Förvaringställe – Where deposited		
Muita tietoja – Övriga uppgifter – Additional information		

Abstract

Machine Translation is ever-present in our globalized world. With the globalization efforts, language mixing during communication, so-called code-mixing, is becoming increasingly more common as well. This linguistic phenomenon always poses challenges for Machine Translation models, to which the answer is often to train bigger models with more data. However, with global warming on the rise, there is a need for models that are less resource-intensive but still maintain a high level of performance. Knowledge Distillation is a tool that has shown effectiveness in reducing model size while maintaining performance; however, research into the uses of Knowledge Distillation in dealing with code-mixing is still scarce.

In this thesis, I explore model robustness of Machine Translation models when dealing with code-mixed data for Spanish-English code-mixing, without being explicitly trained for it, and the effects that Knowledge Distillation has on model robustness for Language Normalization tasks. Student models of three teacher models, as well as baseline models, of different sizes are trained and scored for their performance in a Language Normalization task for code-mixed Spanish-English-to-English. Results are statistically analyzed to determine the effect of Knowledge Distillation on student models compared to teacher and baseline models, both for code-mixed data and individual tasks involved in training, namely Spanish-English translation and English cleaning. Sentence analysis of particularly challenging sentence features is performed for selected sentences to investigate how teacher, student, and baseline models differ in handling these issues.

The experiments show a clear benefit of Knowledge Distillation for code-mix handling even in more compressed models, both when compared to teacher models as well as baseline models. However, there is a significant performance drop with continued model compression. Individual tasks' performance is mostly maintained by student models compared to their teacher which may be better or worse than baseline models depending on their teacher's performance.

Sentence analysis shows an inheritance of teacher model translation tendencies. Students struggle with out-of-vocabulary words as well as spelling errors or colloquial expressions, compared to teacher models. Baseline models often show slightly worse normalization quality but more variability in word choice.

The experiments show that Knowledge Distillation is a useful tool for increasing noise handling even without seeing code-mixing during training; however, in order to conclusively confirm these results, more research has to be done regarding the number of involved languages and general noise handling.

Contents

Chapter 1

Introduction

With the world becoming more connected thanks to the internet and social media, easier travel to different countries and education offering more languages to learn, multilingualism is becoming more and more common (https://www.amacad.org/ and https://www12.statcan.gc.ca/). In addition, code-mixing and code-switching have been common and are becoming a bigger focus for language research as is evidenced by the increase in published papers concerning code-switching shown by Zhong and Fan (2023). Code-switching usually means the switching of languages ("codes") in between sentences, while code-mixing refers to switching languages in the middle of utterances (Setiawan (2023)). The rise of multilingualism, and code-switching and code-mixing being a prevalent linguistic phenomenon, in combination brings with it its own issues due to differences in languages that can influence the evolution of language and communication which can make it difficult to understand one another (even though it has been shown by Kovács (2009) that bilingual children at least are able to infer that their conversation partner might not speak all the languages they speak). Due to this possible complexity of language evolution, the field of NLP needs to be able to be able to make optimal use of research and language modelling in order to keep up and provide appropriate solutions both for research and economically.

Additionally, Large Language Models (LLMs) such as ChatGPT are a significant tool available to millions of people around the globe. Translation models like DeepL and Google Translate are widely popular when translating text on the go and even in professional contexts.

However, these models are often very large and therefore require a lot of space and energy each time they are used and trained. While developers and publishers are aware of the issue and offer solutions such as payments per used token to account for produced CO_2 (such as OpenAI's ChatGPT's pricing listed in https://openai.com/api/pricing/), a more effective and long-term solution would be to have these large models available in a

smaller format while maintaining a high level of performance.

Additionally, translation models are often trained on bilingual sentence pairs for the languages they are supposed to be used for, which, in the case of multilingual Machine Translation, means a possibly high number of supported languages as well as a lot of training data in order to cover all possible language combinations for a given model. This high amount of training data, however, does not include possible code-mixed or code-switched combinations of languages, meaning that, formally, multilingual Machine Translation models are only trained for bilingual translation. It is therefore surprising and welcome, that Johnson et al. (2017) shows that multilingual translation models are able to perform well with code-mixed data even though they were not trained on code-mixed data. However, Srivastava and Singh (2020) showed that conventional translation models such as Google Translate struggle with code-mixing which highlights the complexity of the topic.

Furthermore, multilingual translation models that are able to reliably translate between a large number of languages tend to be very large which also slows down their translation speeds due to the increased number of computations as shown by Jooste et al. (2022). With instant translations for spoken language as well as an expected fast performance speed for written text, models are needed that offer quick, reliable and accurate performance and are small enough to not impact the environment as much as they are currently doing. In addition, smaller models can possibly run on local devices, which provides an additional layer of data safety as the input does not have to be transferred to a server before it is processed.

In order to address the space and resources issue, Knowledge Distillation poses a reliable tool to help smaller student models reach the same performance as, and in some cases outperform, their bigger teacher models. However, the question is whether or not models of smaller size, which have smaller capacities to embed information, are able to keep up with their teacher model's performance and comparable baseline model performance when it comes to handling code-mixed data that the model was not trained on.

In this thesis, I aim to test how well models perform with unknown noisy code-mixed data for a Language Normalization task when they have undergone Knowledge Distillation. I use the Knowledge Distillation pipeline "OpusDistillery" for this, as it offers simple and easy setups for Knowledge Distillation training. Language Normalization tasks require a model to take an input consisting of several languages and transforming it into a text of only one predefined of the involved languages which is a suitable task for this kind of experiment since it requires the models to leverage overlapping embedded information and create a coherent output.

This thesis is divided into several chapters to explain the theoretical basis for my

research. In chapter 2, I will explore the current state of research when it comes to Knowledge Distillation and code-mixing and code-switching. After this, I will shortly go over Machine Translation in general and with a focus on the model architecture that I will be using in chapter 2.1. In chapter 2.3, I will explain the process of Knowledge Distillation before going into the specifics of OpusDistillery in chapter 2.4. With the background covered, I will introduce the data I will be using in chapter 3.1 and the models and setup used during my research in chapter 3. The results of my research are presented in chapter 4, and an analysis and discussion of these results can be found in chapter 5 before presenting the conclusion of my findings in chapter 6.

1.1 Research Questions

My research questions are the following:

- Can performance of multilingual Machine Translation models for Language Normalization tasks with code-mixed noisy data be improved with Knowledge Distillation even though models are trained with bilingual training data?
- Does Knowledge Distillation significantly improve performance of student models when compared to baseline models of the same architecture when trained on the same training data?

In order to answer these questions, I will train several student models based on several teacher models and compare their performances on normalizing Code-Mixed Spanglish-English data. Additionally, I will train baseline models for the same task with the same data the student models are trained on in order to compare the distilled models to how Knowledge Distillation changes the model robustness.

I am expecting to see an increase in performance for at least the transformer model architecture student sets due to the proven effectiveness of Knowledge Distillation shown by Tan et al. (2019). However, I hypothesize that the too much compression of models may not lead to high performance for teacher models which support a vast variety of languages due to the reduced capacity of smaller models, as was pointed out by Mirzadeh et al. (2020) and Huang et al. (2022). I am, however, expecting to see an increase in performance for compressed models compared to baseline models.

Chapter 2

Background

While there is research going on regarding Code-Switching and Code-Mixing in Neural Machine Translation (NMT) models, as well as Knowledge Distillation on NMT models, there does not seem to be research currently that specifically investigates the effect of Knowledge Distillation on models' abilities to deal with code-switching or code-mixing especially when not explicitly trained for it. A lot of research also uses synthetic datasets for Code-Switch data generation, such as Chatterjee et al. (2023), who generated Code-Mixed data based on a system called PACMAN, which they successfully used previously for generating near-naturally existing code-mixed data. They then fine-tuned models with this data and recorded their performance, where it was shown that this approach does increase model performance. However, since my focus is on whether or not Knowledge Distillation is a viable tool that can be used for general MT models without fine-tuning or retraining, this approach is outside of my scope. Likewise, Xu and Yvon (2021) created a synthetic code-switched dataset that is able to help MT models to surpass their initial performance on naturally occurring code-mixing and code-switching. While this approach is equally interesting, I am investigating how models perform with code-mixed data without being explicitly trained on code-mixed data.

Nguyen et al. (2023) showed that MT systems can even achieve higher performances when trained with code-mixed data compared to monolingual input, which is very promising for the field. However, I am interested in how models perform when they are explicitly not trained with code-mixed data. Additionally, the question remains whether Knowledge Distillation can improve these performances further.

There is existing research for the effect of Knowledge Distillation on MT models, most notably Tan et al. (2019) which partially inspired this thesis as well, where multilingual MT models were trained using three different approaches to Knowledge Distillation, with great success: The student models were able to generally maintain or improve their teacher's performance in translation tasks. However, this approach does not include code-

switched or code-mixed data, leaving the question open as to what the effect might be in this case. The study by Pan et al. (2024) showed interesting results; however, as they investigated general noise handling capabilities for translation models for a translation task from Indonesian to Chinese without specific focus on Code-mixing their results do not prove the performance changes due to Knowledge Distillation conclusively. 4 of Facebook's NLLB models were tested and out of those, two models were distilled from bigger models, namely nllb-200-distilled-600M whose teacher model is sadly unkown, and nllb-200-distilled-1.3B which is distilled from the 3.3B model. The researchers performed detailed error analysis for different kinds of noise present in the data, one of which is code-switching. Their experiments showed that the quality of dealing with code-witching declined for the smaller distilled models. However, the teacher model performance is not given for their experiments, which only allows for assumptions regarding the effect of Knowledge Distillation between teacher and student models. It will be interesting to see how models perform as well, if the test set used is specifically designed to contain a large amount of code-mixing or code-switching, since Pan et al. (2024)'s research was focusing on general model robustness.

Johnson et al. (2017) performed experiments for how well translation models are able to handle code-mixing and code-switching without being trained for it. For this task, models will have to leverage overlapping information between languages the models are trained on. Marvellous et al. (2025) performed similar research, specifically focusing on low-resource languages. Both experiments showed that even though performance is generally lower than for non-mixed input, models were able to adapt to the task and handle the requested input with high quality.

As mentioned, there is a lot of research regarding the effect of Knowledge Distillation on MT models as well as model capabilities on code-switching, but as far as I am aware, there is no research yet that specifically targets the effect of Knowledge Distillation on model performance when handling exclusively code-mixed data. While Pan et al. (2024) shows interesting results which suggest that "model distillation may enhance the model's proficiency in specific areas while potentially compromising its capability in other areas" (Pan et al. (2024)), it will be interesting to see whether my research will also suggest this since I focus primarily on one error type that was identified in their research.

Due to the lack of related work for my research, it seems that my thesis presents novel research for the field of Machine Translation as of the point of writing this thesis.

2.1 Machine Translation

Machine Translation is the process of using a computer-powered system to translate a given text from one language into another. The history of Machine Translation (MT) and its approaches is long and varied. In this chapter, I will briefly present steps and achievements MT has made since its introduction.

MT can be broadly categorized differently depending on the scope. There is a categorization based on the methods of the process, which would be rule-based methods and corpus-based methods (Wang et al. (2022)), or based on the approach, which keeps the rule-based models as their own category, but expands the corpus-based methods into example-based, statistical, and neural ones.

Rule-based approaches to MT are the first attempted ones and conceptually were introduced in 1947 (Wang et al. (2022)). The idea is to manually encode grammatical rules for translation of text from one language to another so that, when given a sentence in one language, the model is able to reconstruct the sentence in its appropriate grammatical form in the target language. This approach was largely not scalable, as all rules had to be hand-written and were not easily transferable to other languages. If a translation from or to a previously unsupported language had to be done, this meant long and tedious work. Additionally, as translations systems used bilingual dictionaries for words, creating a multilingual system was extremely difficult due to the inflexibility of the rules.

In contrast to rule-based approaches, corpus-based models make use of an available parallel corpus of sentences in source and target languages and use these corpora to directly have models learn intrinsic grammatical rules for translation (Wang et al. (2022)). The first of these approaches were example-based models. These models identify sentences that need to be translated and use similar sentences from their corpus to create the translations. While able to produce higher quality results than rule-based models, the need to cover as many possible sentences as possible made the approach quite inflexible. Nowadays, this approach is mostly used when manual translators use the support of computers (Wang et al. (2022)).

After example-based approaches to MT, statistical methods were proposed. First introduced in 1990, the idea is for a model to learn grammatical representations of language through the use of statistical methods (Wang et al. (2022)) which would allow for consistent translations which are flexible with different sentences. Due to the eventual success of statistical methods of MT, several improvements have been implemented over time. These include, but are not limited to, models that make use of morphological information and models that work with syntactic parsing trees (Wang et al. (2022)).

While statistical models further improved on MT quality, the increasing progress in neural network and deep learning research led to statistical models being replaced for MT

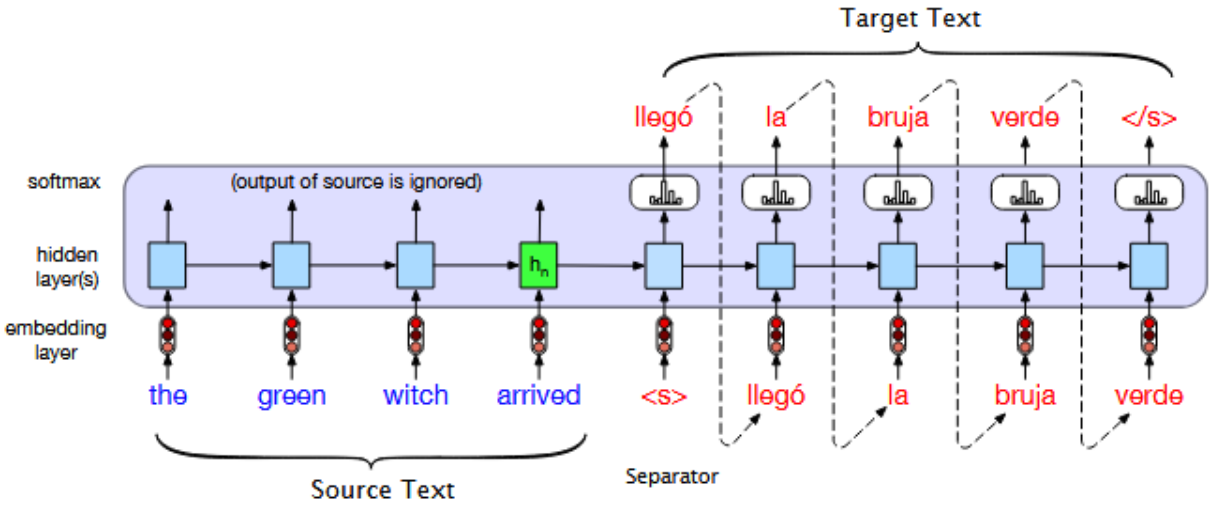


Figure 2.1: Seq2Seq Model Architecture - Source: Jurafsky and Martin (2025)

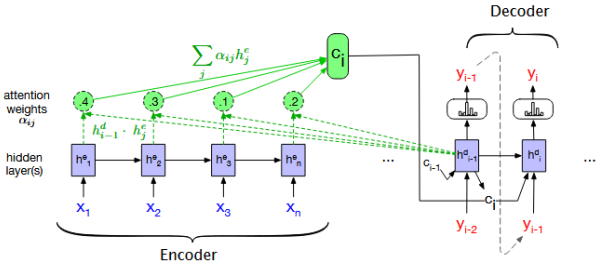


Figure 2.2: Attention Mechanism of Seq2Seq models - Source: Jurafsky and Martin (2025)

tasks with the arrival of sequence-to-sequence (seq2seq) neural models. These models, proposed in 2014 by Bahdanau et al. and Sutskever et al., make use of two recurrent neural networks, one acting as an encoder that embeds the information of the source sentence into a fixed-length vector, the context vector, and one that decodes this information to generate the corresponding target sentence. An example of these seq2seq models is shown in figure 2.1 as it is presented in Jurafsky and Martin 2025. Seq2Seq models further improved translation quality due to the versatility and power of neural networks. The seq2seq model was further enhanced by a so-called attention mechanism, shown in figure 2.2. This addition made it possible to teach models what word to focus on in a given position in the generation of the target sentence and further improved upon the system. The seq2seq architecture and especially the attention mechanism laid the groundwork for what is nowadays widely used for MT tasks everywhere: The transformer.

Transformers were introduced by Vaswani et al. (2017) and use a similar architecture to seq2seq models, as in, they use an encoder part which embeds the input sequence and a decoder part which uses the embedded information to generate the next token. The architecture as it was introduced in the original paper is shown in figure 2.3. However,

instead of going through an input sequence token by token and embedding it, then adding the embeddings of the next token and so on until the context vector is created and then using this vector and the attention scores for the current position, the transformer embeds the entire input sequence at the same time and passes it to the self-attention mechanism, which is the driving force of the transformer. Self-attention allows for much more nuanced assessment of the current position in a sentence as well as which parts of the input sequence are important to focus on at the given position and what context to take into account than the previous seq2seq models were able to do. The mechanism itself consists of three weight matrices which are trained during the training process, the key, query and vector matrices. The input is then matrix-multiplied with each of these matrices and the key and query vectors are matrix-multiplied to form an attention matrix which signifies which part of the sentence so far at any given position in the input sequence has the most importance for the output generation. To this attention matrix, a softmax function is then applied to normalize its values and it is matrix-multiplied with the value matrix in order to create the output for one so-called attention head. A transformer model's self-attention mechanism can consist of several attention heads; the original transformer, for example, consists of eight such heads. The heads' outputs are then concatenated and projected back to their original shape from when the input sequence was embedded. This way, the output of the self-attention mechanism can be summed up element-wise with the original embeddings and passed on to the next part of the model. This can be either another encoder part or a decoder part. The decoder matrix multiplies the resulted matrix consisting of embeddings and self-attention outputs with its own weight matrix to pass the outputs to the decoder stack activation function. After this step, the outputs are projected back to their original shape again to be summed up element-wise with the initial matrix of the decoder block. This step can be repeated several times depending on the number of decoder blocks. Finally, the matrix is used as input for a linear layer and matrix-multiplied to get a final matrix which, when softmaxed, can be used to select the next token for the output generation. At this point, the currently generated sentence is passed through the entire decoder stack again to predict the next token until the predicted token is the end of a sentence and the generation step ends.

The entire process of a transformer processing an input and generating an output can be followed in a visualized format in the https://scillidan.github.io/transformer-explainer/ and in even more detail in the https://bbycroft.net/llm. Since the models I will be working with are transformer architectures it is important to explain how these models work in detail.

With Machine Translation as prominent as ever, they also revolutionized a topic which has always been problematic for MT: Multilingual Machine Translation.

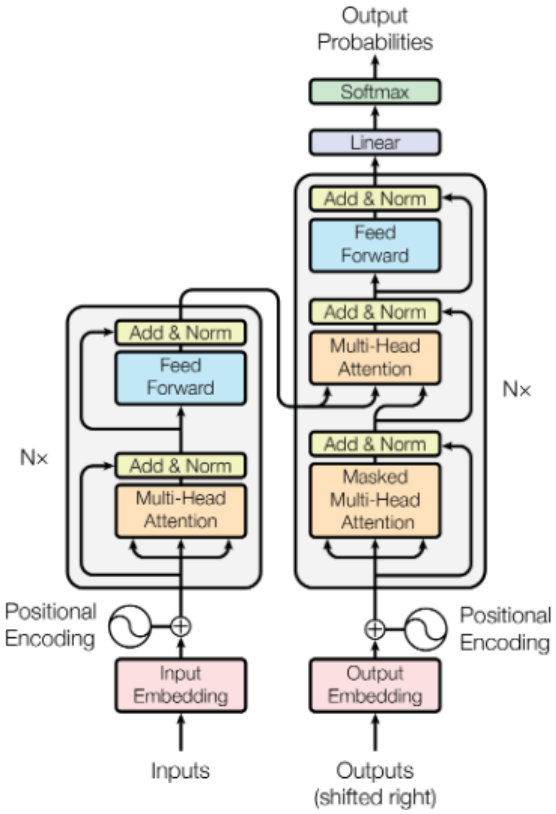
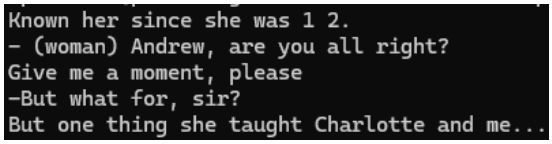


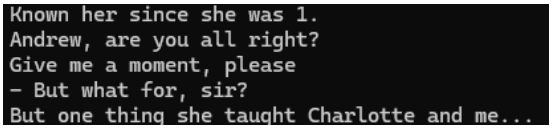
Figure 2.3: Transformer architecture as introduced by Vaswani et al. (2017)

2.2 Multilingual Machine Translation

Multilingual Machine Translation has always posed unique challenges for MT models. Most obviously, languages differ widely in terms of grammar, sentence structure and of course vocabulary. Secondly, in order to train a successful translation model, a parallel corpus is needed in order to provide the model with sufficient input and respective output sentences. However, for a lot of language pairs, there are not enough parallel corpora available, if any (Wang et al. (2022)). This issue can be leveraged and has been addressed by different methods, even before transformers and neural approaches. The most prominent one is the use of an intermediate language for translations for which a parallel corpus might more likely exist. Wang et al. (2022) uses an example of a translation from Chinese to German, which uses English as the pivot language. The Chinese input is first translated into English, and the English output is then translated into German since the parallel corpora for these languages are much more common and rich than Chinese-German ones. With the arrival of neural methods, however, this situation changed as a single model could learn to translate from and to several languages. Even smaller corpora for language pairs are not as big of an issue anymore as before since models can use overlapping information of linguistically close languages to improve performance on so-called



(a) Training corpus source example



(b) Training Corpus target example

Figure 2.4: Example from en-en training corpus for teacher model 1 students and baseline models

"low-resource" languages (Wang et al. (2022)). Since this thesis will focus on code-mixed sentences, having models that are robust with multilingual translation is very important, which is why the transformer architecture is perfect for this task. Since several languages are unifiedly available in a single model, several different ways to perform multilingual MT are possible with the same model: one-to-many, many-to-one, and many-to-many. Because in all of these model types the weight matrices used for the computations are shared for all suported languages, the model becomes able to not only translate between the supported languages, but also to infer embedded knowledge from languages in order to improve translation quality and even deal with code-switching due to utilizing overlapping information of languages as shown in Johnson et al. (2017).

One-to-many translations describes when a MT model supports one source languages which can be translated into a number of target languages. This is usually managed by introducing a special token at the end or the beginning of the source sentence which indicates the language the sentence is supposed to be translated into. Many to one translations are the opposite, where a number of source languages can all be translated into a single target language. These are the kinds of models I will be training for this thesis. Finally, a many-to-many model combines the two approaches by being able to translate from one set of languages to another set of languages. These models again use special tokens to identify the target language for a sentence and can possibly use more special tokens for source languages.

In order to train a Machine Translation model properly, one of the most commonly used forms of datasets are parallel corpora. These dataset formats typically come in the form of sentence-aligned plain text files with translated sentences on corresponding lines, one for the source sentences and one for the target sentences. An example of this is shown in figure 2.4, which shows the first 5 sentences from the source and target corpus used for teacher model 1's students and baseline models.

Using this setup, a variety of models can be trained and the data easily set up. In case of a bilingual translation model, a simple source file consisting of the input language sentences and a simple output file consisting of the target language sentences is enough.

In case of a many-to-one model, the source languages might exist in several files, but

can simply be merged into one source file containing all sentences. The target file will in this case contain the corresponding output sentences in the same order to preserve the sentence pairs. In the case of using the same sentences across all languages, this will lead to copies of the same sentences being present in the target file. This does usually not impede training, but is important to keep in mind when it comes to dataset cleaning.

In case of a one-to-many or a many-to-many model, the setup is largely the same, all languages can be merged into the same files, however the fact that the model needs to learn to differentiate between target languages makes it necessary to perform additional preprocessing of the source sentences. Usually, as mentioned, this consists of adding a language specific token to the input sentences which indicate the target language for the sentence. This also makes it possible to perform dataset cleaning without deleting the possible copies of sentences if the source sentences share languages.

As my thesis will focus on training many-to-one models, the training and test data do not need these target language tokens, even though the pipeline used for training the student models, named OpusDistillery and introduced in chapter 2.4, supports it.

2.3 Knowledge Distillation

As MT models and deep neural networks in general tend to be very large and use a considerable amount of energy for training and inference, a method has been proposed to compress a bigger model into a smaller, faster and more energy-efficient model (Hinton et al. (2015)). This process is called Knowledge Distillation (KD). It utilizes the knowledge embedded in a big teacher model and teaches it to a smaller student model. A benefit, next to energy efficiency, is, that a student model, trained with a teacher model tends to train more quickly than the teacher model counterpart and sometimes even models of the same size as the student models which were trained from scratch (Jooste et al. (2022)). Moreover, student models also perform faster than their teacher counterparts (Jooste et al. (2022)).

There are several ways to perform Knowledge Distillation, which can broadly be categorized into three groups (Gou et al. (2021)): Response-based Knowledge Distillation, Feature-based Knowledge Distillation, and Relation-based Knowledge Distillation.

Response-based Knowledge Distillation focuses on the outputs from teacher models to guide the student training process. This could be a translation of the training samples, also known as Sequence-Level Knowledge Distillation, or the direct softmaxed outputs from the teacher model for every single token during inference, also known as Word-level Knowledge Distillation. Both of these methods will be explained in further detail in sections 2.3.1 and 2.3.2.

Feature-based Knowledge Distillation goes a step further than response-based Knowledge Distillation and focuses on the intermediate layer outputs of the teacher and student models to guide the student training. This approach, however, has not been researched sufficiently for wide-range use, as the difference between the sizes of the teacher and student layers, which have to be compared, can vary greatly and information can not be easily and meaningfully transferred from bigger output vectors and smaller ones. In comparison, word-level KD only requires student models to have the same target vocabulary for generating output tokens as the teacher model has. This allows for way more flexible model design since only the very last step of student models has to be directly comparable to the teacher models.

Relation-based Knowledge Distillation takes feature-based Knowledge Distillation even further by also taking into account the relation between training samples and how the interpretations of these samples differ between student and teacher model. This approach, just like the feature-based approach, needs more research to be widely useable as the modelling of samples provide an additional layer of complexity in addition to the transfer of large vectors to smaller ones.

Since response-based Knowledge Distillation is the most researched of the approaches, I will explain these in further detail in the following sections. Especially sequence-level Knowledge Distillation is important as my thesis focuses on this approach.

2.3.1 Word-Level Knowledge Distillation

Word-Level Knowledge Distillation uses the power of the bigger teacher model to guide the training of the student model on a token-by-token basis. Since for any seq2seq model, the decision for which token to place is made using a softmax function which during training is aiming to become a 1-hot vector, this approach to Knowledge Distillation aims to soften the hard requirement of the 1-hot vector by aligning the student output instead with the teacher model output (Hinton et al. (2015), Kim and Rush (2016)). This leads not only to more lenient training but also makes it possible for the teacher model to influence each output step of the inference process for an output during training. As the student model is attempting to imitate the teacher model, it learns how the teacher would decide in the given situation, thus enabling the transfer of the embedded knowledge of the teacher model.

While this approach to Knowledge Distillation provides very direct transfer of knowledge, there are a few drawbacks that have to be taken into account. For one, since at each prediction step we not only need the predictions from the student model, but also the teacher model, this can add considerable time to the training time and resources that are required. Additionally, as each word is being estimated individually by the student

model and not in direct context of the embedded sentence, even though the teacher model predicts tokens in context of the sentence, this can affect understanding negatively due to the shallow prediction of tokens.

There are variations of this approach, as presented in Kim and Rush (2016) in which it is possible to choose whether to use the classic 1-hot approach or the teacher distribution, or where instead of the entire distribution the student model only tries to fit its predictions over the top-k scores of the teacher model. These approaches show promise as presented in Kim and Rush (2016), however, will not be included in the scope of this thesis due to the current limitations of the used pipeline OpusDistillery.

2.3.2 Sequence-Level Knowledge Distillation

In contrast to word-level KD, sequence-level KD focuses on aligning student predictions to teacher predicted sequences instead of individual tokens. This is achieved by turning the training into a two-step process: The training set is first translated by the teacher model. The resulting new sentence pairs are then used for a normal training process for the student model (Kim and Rush (2016), Tan et al. (2019). This allows the teacher model to guide the student's training with a bigger focus on sentence context than is possible than with word-level KD.

This approach allows the teacher model to also filter possibly problematic or difficult translations by creating an output that is machine-translated and therefore easier to interpret by MT systems.

2.4 OpusDistillery

In order to introduce OpusDistillery, which can be found on its GitHub page https://github.com/Helsinki-NLP/OpusDistillery, it is necessary to introduce the sources upon which the tool is built. OpusDistillary uses the "Firefox Translations Training pipeline" (FTT) (the project's Github can be found at https://github.com/mozilla/translations) which itself was developed as part of the Bergamot project which can be found under https://browser.mt/. The Bergamot project itself was developed to make translation tools available locally in a browser for translating entire websites on your local machine without the need for an external server to handle the translations. FTT focuses on creating the Firefox browser translation extension using this pipeline in order to efficiently train quick and small models which can be used for local browsing.

Opus Distillery adapts the FTT pipeline in order to be used with Opus-MT models, although by now also huggingface teacher models are supported. In addition, GPU utilization reports can be accessed in order to monitor energy consumption during training

and to be able to make the entire process more energy efficient. Furthermore, the pipeline added support for multilingual models, making it possible to use any number of bilingual as well as multilingual teachers and training bilingual and multilingual students. Due to the compatibility with Opus-MT models, the possibility of multilinguality and the streamlined training process which can be fully utilized on the University of Helsinki's supercomputer Puhti, I chose OpusDistillery to perform the training of my student models.

The pipeline offers a wide range of options to customize the training process, as well as data download and preprocessing, which will be dicsussed with regards to this thesis in section 3.2

In the following, I will present the steps the pipeline takes and explains them in more detail. These steps are taken directly from the OpusDistillery documentation at https://helsinki-nlp.github.io/OpusDistillery/pipeline/steps.html. From configuration files for slurm jobs for Puhti, the training and model parameters and dataset and language specifications the pipeline builds a Directed Acyclic Graph (DAG) using these steps which includes everything needed from downloading the datasets and teacher as well as backwards models for scoring teacher translations to training, evaluating and exporting the final student model. During the pipeline process this DAG is being updated constantly with steps becoming necessary, such as translating a dataset after downloading, cleaning and splitting it into files of specified size. The full automation of the training and knowledge distillation process is what makes this pipeline so powerful to use.

The training framework used by the pipeline is MarianNMT, which is a frameowrk used for training Machine Translation models with a focus on quick and efficient training. MarianNMT is further discussed in section 2.5

However, the pipeline currently only supports sequence-level Knowledge Distillation due to technical limitations. Because of this, and because the training process for the student models should be as uniform as possible to enable comparison of the model performances, my thesis will focus on sequence-level Knowledge Distillation only.

The steps OpusDistillery uses during the training pipeline are the following:

- Installing Dependencies - Making sure all dependencies are installed and compiled correctly
- Data downloading - Downloading the specified dataset for training with the amount of sentences specified in the configuration files
- Data cleaning - Preprocessing datasets, following the steps included in the config file for OpusFilter (further discussed in section 2.6) in order to make sure data is noise free
- Merge and dedupe - Merge cleaned datasets and deduplicate samples

- Training vocabulary - Using SentencePiece (further explained in section 2.7) trains a vocabulary and tokenizer model for student models based on the corpus
- Teacher download - Download teacher model or teacher models in case of ensemble training
- Backward model download - Download backwards model (in my case the same as the teacher model with switched input and output capabilities) for scoring translated sentences
- Translation by teacher - Teacher model(s) translate the training corpus to create the new gold label sentences for knowledge distillation training
- Cross-entropy filtering - Filtering by the backward model by using the marian-scorer of the translations and removing the sentences with the lowest scores to create higher quality training set
- Training alignments and shortlist - Using fast_align (the project's GitHub can be found under https://github.com/clab/fast_align) train word alignments and extract lexical shortlists for exporting the final model to Bergamot format for quick and CPU-intended use
- Training student - Using the newly translated training data, train a student model according to the configuration files. This file contains information on the intended architecture and size of the student model as well as the training parameters. There are baseline files which provide parameters which are not specified in the experiment configuration file in order to guarantee a smooth but highly customizable training experience
- Fine-tuning student - Fine-tuning the final student model to specific languages used during training to obtain specialized students
- Quantization - quantizes fine-tuned student models using 8bit quantization to evaluate the model on CPU. For doing this, memory and storage requirements are being reduced by limiting model weights representation to 8bit integers instead of 16 or 32bit (definition taken from https://huggingface.co/docs/transformers/main/en/main_classes/quantization)
- Evaluation - Using SacreBleu, a tool available on Puhti for scoring parallel corpora, calculate metrics for all trained and fine-tuned models. Metrics included are BLEU and chrf. While BLEU will be explained in section 2.8 in more detail as it has

significance for this thesis, chrf is a metric which breaks down words into character n-grams and calculates the overlap between the prediction and the gold label sentences. This produces a score which shows a model's ability to translate text by placing particular importance on word stems instead of whole words.

- Export - Export the trained model and shortlist to Bergamot translator format to be used as intended from the Bergamot project

2.5 MarianNMT

MarianNMT is a Machine Translation framework designed for fast performance. It was introduced by Junczys-Dowmunt et al. (2018) and is being developed mainly by the Google Translate team as well as independent researchers and contributors. MarianNMT is also the current back-end for Google Translate, showcasing its efficiency for commercial and professional use. It is hosted at https://github.com/marian-nmt/marian where installation instructions can be found as well.

To achieve its fast performance, it is written in C++ and is able to perform both on CPUs and GPUs, with GPUs offering a significantly higher training and inference speed. For translation model architectures, MarianNMT offers deep Recurrent Neural Networks (RNNs) and transformers.

In Junczys-Dowmunt et al. (2018), MarianNMT's superior training speed is shown, outperforming Sennrich et al. (2017) translation models trained using the Nematus framework, by training 4 times as fast on a single GPU.

MarianNMT comes as a compiled build that takes a config file with the required parameters, such as encoder-decoder layer size, embedding dimensions, feed forward layer size etc, for training a translation model as well as the training parameters, such as mini-batch size, early stopping settings and optimizers with their respective options.

A trained model offers several ways of inference, choosing either to focus on the separate final model and vocabulary files, or a summarized decoder.yaml file, all of which are part of a finished model.

MarianNMT is the framework used in the OpusDistillery pipeline, meaning the configuration files for training models include the necessary parameters for training a student model in the Marian format. In addition, the pipeline offers some default settings for training and translation of models which are overwritten by the settings given by individual config files. This allows for nuanced training as only the parameters which are changed from their default state need to be included in the config file.

In the configuration files for OpusDistillery the model and training parameters are listed under ”training arguments” and ”marian-args”.

2.6 Opusfilter

OpusFilter is a tool for filtering as well as scoring data for Machine Translation models in preparation for training. It was introduced by Aulamo et al. (2020) and offers a wide variety of possible filters and scoring options.

Filters can be divided into roughly two groups, rule-based filters and trained classifier models, the latter of which relies on trained neural networks to classify sentences based on the given arguments. This is obviously more resource intensive than a rule-based approach. OpusFilter has demonstrated its usefulness in improving model performance by reducing the size of a training set while removing noise, both for rule-based approaches and classifiers. Additionally, OpusFilter offers custom filters and scores for an even more specialized filtering of training data.

The tool takes a yaml-style configuration file which includes all information about the rules to be used.

OpusFilter is featured as a step in the OpusDistillery pipeline, both as an option for custom filtering and as part of a set default script, which is what I used in my experiments. The used filters are listed in chapter 3.2.

2.7 SentencePiece

SentencePiece is a tool for automated vocabulary creation for NMT models. It is hosted under https://github.com/google/sentencepiece, although Google does not claim ownership of the tool itself.

SentencePiece offers automated tokenization of texts using different segmentation algorithms, most prominently ”Byte-Pair Encoding” (”BPE”) and ”Unigram Language Model”(”ULM”), which is essential for creating vocabularies for Machine Translation models.

As MT models need to be able to make predictions for words during their decoding part of creating translation predictions, but also encode input text properly, model vocabularies need to be able to include as many different words and word forms as possible.

However, words can take different forms due to differing morphemes corresponding to use cases and grammatical cases in their respective languages, which means creating a vocabulary for even a small number of languages can become exponentially big if it contains only full words. Consider this example from a language with rich morphology, such as German. The German word ”schlafen” (eng. ”sleep”) would need 15 separate vocabulary slots to cover all its forms (”schlafen”, ”schlafe”, ”geschlafen”, ”schläfst”, ”schläft”, ”schlaft”, ”schlief”, ”schliefst”, ”schliefen”, ”schliefet”, ”schliefe”, ”schliefest”, ”schlaf”, ”schlaft”, ”schlafend”).

Segmentation algorithms such as BPE and ULM aim to reduce the size of vocabularies while still accounting for as many morphological forms as possible by segmenting words into so-called subword tokens. By making a model tokenize raw text into these subword tokens and having the model predict these subword tokens instead of whole words, MT models are able to account more easily for different grammatical forms and cases of words while keeping the sized of their vocabulary manageable.

Of course, it is possible to reduce the size of the vocabulary by simply having the model predict each letter by itself which would mean for an English vocabulary the vocabulary would include the alphabet, numbers and special characters, reducing the size of a possibly multiple millions encompassing size effectively to under 100, or even only 26 if only letters are involved.

However, this also means that the model predictions would become more volatile as there are more possibilities for confusion due to the number of possible combinations of these 26 letters. The task of segmentaion algorithms such as BPE and ULM is therefore to balance the size of a vocabulary (which is usually given as a hyperparameter) and the actual efficiency of individual so-called "sub-word" tokens.

Consider again the example of a vocabulary that consists of the possible forms of "schlafen". While at first, a vocabulary of 15 slots would be needed to express all possible word forms for this vocabulary, BPE and ULM might end up with a segmentation which includes splits "schlafen" into "schl-a-f-e-n", or "schliefst" into "schl-ief-st". These segmentations also cover the separate forms "schlaf", "schlafe", "schlief" and the subword tokens can even be reused for different words. "schl" can be reused for words like "schlecht", "schlagen", "ief" can be used for words like "tief", "rief" or "triefen". Moreover, this way common affixes and suffixes, such as the affix "ge-" from "geschlafen" can be identified automatically and reduce the needed number of slots drastically, as now words that take a form which includes "ge-" do not need their own separate slot anymore.

With SentencePiece, different vocabulary styles can be trained. Vocabularies for input and output can be kept separate, or, by concatenating source and target files, a single vocabulary for input and output can be created. This might be especially useful when source and target languages share languages or when datasets are very small in order to obtain a good representation of segmentation across all the data.

SentencePiece is part of OpusDistillery, where it is used to create a unified vocabulary file for source and target languages. This is the way I use the tool for my experiments.

In the follwing sections I will introduce BPE and ULM in more detail.

2.7.1 BPE

BPE, originally introduced by Gage (1994) as a compression algorithm, is an unsupervised tokenization algorithm that can be applied to raw text in order to create a vocabulary which is useable by MT models. In its currently widely used form, it was introduced by Sennrich et al. (2015). It works by first segmenting the dataset into individual letters and then iteratively builds the vocabulary by identifying the most common letter pairs and merging them into one. By building letter sequences based on the most common letter or sequence pairs, it can be assumed that these sequences will be common in unseen data and rare words. This process can be repeated as many times as needed until the desired vocabulary size is reached.

2.7.2 ULM

ULM was introduced by Kudo (2018) and uses a similar approach to BPE. However, it works with a probabilistic model to estimate the impact that the removal of a letter sequence would have. By first building a big vocabulary which includes single letter segmentation and common subwords from the corpus and incrementally reducing the size of this vocabulary until the desired vocabulary size is reached, it achieves a similar quality of segmented vocabularies. However, as BPE works only greedily, meaning the most common pair is always chosen for merging, ULM might select less common sequences to keep as their impact if they were removed from the vocabulary would be too big.

2.8 BLEU Score

BLEU score is a metric introduced by Papineni et al. (2002) and widely used in scoring MT sentence predictions in a quick and simple way. The idea is that a correct translation of a sentence should share a lot of words with the corresponding gold label sentence. This means that a similarity score between a predicted sentence and the target sentence can be computed which is what the BLEU score does. It calculates the token overlap for a set number of ngrams between the prediction and the target corpus and applies a brevity penalty in case the model translates a sentence only partially in order to not skew the final score.

While the BLEU score is quickly and easily useable, it also comes with its own set of limitations. For example, representative examples are mostly achieved when applying BLEU scoring to entire corpora so a sentence-level evaluation of translations is not viable for this system. Moreover, since the score does not assess the grammatical correctness or semantic correctness of a translation, but only its literal overlap with the gold label

sentence, it is unable to account for variability in human language as any word that is chosen differently from the gold label sentence is immediately penalized and lowers the score of the translation.

So while the score is very useful to assess model performance for translation tasks, it is important to keep in mind that the score possibly does not capture all the model's translation quality.

Chapter 3

Experimental Setup

3.1 Datasets

The biggest difficulty when finding datasets for the purpose of this thesis was finding proper code-mixed test data to use. Since natural code-mixing datasets are rare as it is, finding the correct parallel corpora which have not only natural code-mixing but also the corresponding language-normalized sentences was very difficult.

While there are approaches how to create synthetic code-mixed data which mimics naturally occurring code-mixing, and with good results as shown by Chatterjee et al. (2023) and Xu and Yvon (2021), it sadly cannot be a 100% replica of natural code-mixing. Additionally, it is possible to create synthetic parallel corpora from existing code-mixed datasets using either manual translations or an existing MT model. Sadly, manual translations are not possible due to limitations of resources and time for this thesis. Machine translation is a more viable and readily-available option; however, as the newly created gold sentences of these datasets would be as machine-translated as the final model predictions, this would create a possible bias in the scoring of the translations which renders the results not representative. Table 3.1 shows how performance changes when scoring translations against a machine translated output and when scoring it against the natural output. The model used for the translations was the publicly available huggingface model Helsinki-NLP/opus-mt-es-en which can be found under https://huggingface.co/Helsinki-NLP/opus-mt-es-en and the tested data is the code-mixed test data which is used for the experiments. The model for translation was chosen because it is trained for translation from Spanish to English without code-mixed data just like the tested models. The tested models are the ones which are used as teacher models for the experiments and will be introduced in more detail in section 3.4. The models are named in accordance to the naming scheme introduced in the same section. The BLEU scores were obtained by using the website listed in Aguilar et al. (2020) where the dataset is available and where the

evaluation of the translated sentences takes place. The dataset will be introduced in more detail in section 3.1.2. The BLEU scores in the right column were obtained by translating the test set sentences with the mentioned Helsinki NLP model and using these translations as the gold sentences. The table shows an immense increase in BLEU score after scoring the model predictions against machine translated gold sentences. Due to the bias these scores would create, the approach of creating my own gold sentences was not an option.

Model	BLEU for original output	BLEU for MT output
Model 1	40.18	51.8
Model 2	42.92	50.8
Model 3	43.62	53.0

Table 3.1: BLEU scores for code-mixed test data when scored normally and against a machine translated version

As a result of my research, I have found two parallel corpora that include naturally occurring code-mixing and the corresponding language-normalized sentences in a Moses format.

One of these, introduced in Dhar et al. (2018), consists of 6,097 sentences which were manually translated. However, since the data is saved using Latin characters and not Devanagari, it makes it impossible to use this dataset for Opus models which can translate Hindi, as the Devanagari character set is needed for it. Therefore, the dataset was excluded from further research for this thesis.

The test set which is ultimately used for my experiments was introduced in Aguilar et al. (2020) and consists of 15,000 training sentences and 6,500 test sentences with Spanish-English code-mixed sentences and their corresponding normalizations. The dataset is detailed in section 3.1.2.

3.1.1 Training Data

The training data for student and baseline models is available on the Opus GitHub under https://github.com/Helsinki-NLP/Tatoeba-Challenge/blob/master/data/README.md and only needs to be preprocessed. I am choosing the same traning data versions that the teacher models were trained on for their respective student models. This means that the student and baseline models of teacher 2 and 3 are trained with the same version of training data while the students and baseline models of teacher 1 are trained with an earlier version of the training set. I chose to do this in order to eliminate the possiblity of differences in the training data influencing the student and baseline model performance as they are supposed to represent a smaller compressed version of their big teacher models.

An entirely same training data set is sadly not possible due to storage limitations which is why the student models are trained for only the languages which are involved in the code-mixed test set, which is Spanish and English. Furthermore, backtranslated data as well as bible data, both of which will be explained in more detail in this section, are also excluded. This means, that the student models, while trained by and using a subset of languages their teachers are trained for, might show different performances for tasks than students which are a true compressed version of the teacher model would.

The datasets used for training during the pipeline are part of the Tatoeba Challenge Set presented by Tiedemann (2020). It does not contain code-mixing or code-switching, meaining my models will be trained as standard multilingual models for a Language Normalization task which consists of code-mixed source sentences. This leads to models not explicitly knowing code-mixing and code-switching and having to infer this knowledge from shared embedded information for the Language Normalization task.

The training data is a compilation of different datasets collected for the OPUS project found at https://opus.nlpl.eu/. The sentences are already shuffled. For this thesis, I have chosen the Spanish-English and English-English datasets in order to account for tokenizing, the process of which is covered in section 2.7 in more detail, and to mimic the code-mixed test set I am using to evaluate model robustness. While the Spanish-English dataset is self-evident, the English-English dataset requires more explanation. The dataset consists of sentence pairs that, on the target side, either retain the same words and structure, in other words, the sentence is being kept as it is, or has some slight paraphrasing, while on the source side the sentences contain some form of noise, such as extra words in parentheses or extra characters that are not present in the target sentence. The dataset, therefore, presents a mixed task that trains models to deal with a certain level of noise in source sentences while keeping the original sentence as intact as possible as well as paraphrase and find synonyms for source sentence words, which will most likely have a positive effect on student and baseline noise handling as well.

The languages were chosen in order to account for the involved languages in the code-mixed test set, namely Spanish and English on the source side and English on the target side. While using datasets such as Spanish-English and English-Spanish would be possible, using them might influence model performance and make it harder to investigate the effect of Knowledge Distillation versus the effect of overlapping language information. However, this dataset would need to be included in order to create a student model that is more similar to its teacher than the ones trained for this thesis.

Each student model's training set will also be taken from the same date collection their teacher model's set was taken from. The Tatoeba challenge set offers 3 different dates of data collection. The teacher model names reflect the date of the data collection

in their names, which is explained in section 3.4.

In order to save space and to account for the smaller model sizes compared to the teacher models, I have limited the maximum number of parallel sentences to 20M per language pair. I detail the exact size of the training and test sets in table 3.2 for each student model set. As can be seen, the en-en datasets are very small after preprocessing for the 2021 version of the dataset. This, in large part, is due to the deduplication step during data preprocessing. After testing with a simple deduplication script, out of 20,000,000 sentences only about 8,600,000 sentences remained. Coupled with the other filters that are applied before and after the translation by the teacher model, it make sense why the dataset is as small as it is. This, unfortunately, means that the students and baseline models that are trained with the 2021 dataset are highly skewed towards the es-en translation side and leave the en-en side very underrepresented. This might also have effects on student and baseline models' abilities to deal with English input.

Dataset	Language	Before Preprocessing	After Preprocessing
Teacher 1 Students	es-en	20,000,000	18,006,836
	en-en	20,000,000	1,890,185
	Total	40,000,000	19,897,021
Baseline Models 2021 Version	es-en	20,000,000	19,429,297
	en-en	20,000,000	2,193,764
	Total	40,000,000	21,623,061
Teacher 2 Students	es-en	20,000,000	17,792,503
	en-en	20,000,000	12,889,806
	Total	40,000,000	30,682,309
Teacher 3 Students	es-en	20,000,000	17,801,750
	en-en	20,000,000	12,958,624
	Total	40,000,000	30,760,374
Baseline Models 2023 Version	es-en	20,000,000	19,251,391
	en-en	20,000,000	16,406,049
	Total	40,000,000	35,657,440

Table 3.2: Training Set sizes before and after preprocessing

3.1.2 Code-mixed Test Data

The code-mixed test data used in my experiments is part of the Linguistic Code-Switching Evaluation (LinCE) benchmark proposed by Aguilar et al. (2020). The benchmark aims to provide a centralized benchmark for comparing model performances for different code-switching tasks.

The tasks included in this benchmark are Language Identification, Part-of-Speech-Tagging, Named Entity Recognition, Sentiment Analysis, and Machine Translation. While

Machine Translation was not an official part of the benchmark by the time the original paper was released, it has since been added and is available for use at https://ritual.uh.edu/lince/home. Unfortunately, as of 01.10.2025 the website is not available anymore, which makes a reproduction of the experiments not possible. Unfortunately, the gold sentences for the dataset have also not been able to be shared with me and I only have the Spanglish training and test data available.

The data for the Machine Translation task that I am using for this thesis was sampled from Solorio et al. (2014) and Molina et al. (2019) which in turn were sampled from Twitter. The data was cleaned to not include any personal account names anymore and in sum contains 15,000 training samples and 6,500 test samples. While the LinCE benchmark does also include code-switched data for Hinglish to English translations, this shows the same issue that the Hinglish dataset from Dhar et al. (2018) had, as the sentences were written in Latin characters which would conflict with the teacher models. Additionally, LinCE benchmark offers Modern Standard Arabic-Egyptian Arabic to English or Spanish translation data. However, for the scope of this thesis, I am only using Spanglish data.

Since the LinCE benchmark offers an online scoring of model predictions, the gold sentences are sadly not available for downloading for any of the provided datasets. Rather, model predictions have to be uploaded in a specific format and the online pipeline performs the scoring automatically. The scores can be shared on the leaderboards which can be found on the website as well. Because of this, the training sentences can not be used for model training, fine-tuning or evaluation as the website throws an error if the prediction file has more or less than 6,500 lines.

Because of this, I have to use the dataset independently of the OpusDistillery pipeline, since it is not possible to include the submission of predictions in there. The submission also only supports the 6,500 test sentences and not the training sentences, which means my code-mixed test set consists of 6,500 Spanish-English code-switched sentences.

Sentences range from rather common sentence structures such as

> <user> yesss we can probs pick you up si es que we convince him

to more colloquial and informal language with lots of chat and social media features, such as acronyms, emojis, or hashtags, as can be seen in this example:

> Flowers day !! 🌸💐🌺 #salhuaclothing Pedidos al 096.880.7384 http://t.co/Cruc6gCN4c

This presents an additional noise challenge in addition to the code-mixing. Sadly, I will not be able to perform cleaning on the test dataset due to the mentioned unavailability of gold label sentences. While the additional noise will certainly affect the general performance of the models, during my detailed sentence analysis of selected sentences, I will focus on the actual translations of Spanish tokens in the sentences.

3.2 Data Preprocessing

Data preprocessing is done by the OpusDistillery pipeline as it involves both data downloading and preprocessing. The preprocessing uses a combination of Perl scripts, Python scripts, shell scripts, and Opusfilter to deduplicate the dataset, normalize whitespace, and apply filtering to reduce noise.

Of these, OpusFilter is the step where I am able to influence the preprocessing steps the most. Since the Tatoeba challenge set is very clean already, there is not much preprocessing necessary. I therefore only apply the default setting which is part of the pipeline. These filters include:

- A length filter to filter out sentence pairs in which a sentence is either shorter than 3 or longer than 100 words
- A length ratio filter to filter out sentence pairs in which one sentence is three times the length of the other.

After translating the training sentences the marian scorer gives each sentence a score and a cross entropy filter is applied to the scored sentences which removes the 5% of the lowest scoring sentences to ensure that only the best translated sentence pairs remain.

3.3 Training

Training of the models is part of the OpusDistillery pipeline and is performed on the University of Helsinki's Puhti supercomputer, which utilizes Tesla V100-SXM2-32GB GPUs for training. Over the course of my thesis, I am using between 1 and 4 GPUs for training due to technical difficulties. The detailed training parameters for the marian training setup can be found in the list below as well as the train logs for each student and baseline model. As the models were trained in the same way, they do not differ in their setup except for the architecture depending on the group of student and baseline models, used training datasets, and the seed used for randomizing the sentences during training.

- precision float32
- save-freq 5000
- early stopping 5 with perplrexity as measure
- learn rate 0.0003
- learn rate warmup 16000 batches

- optimizer adam, beta1 0.9, beta2 0.98, bias $1e^{-09}$
- mini-batch 1000
- seed 0 (random initialization)
- dropout 0.1
- loss function cross-entropy, weighted 0.1
- beam size 1

During training, the development set as well as the test set for each of the datasets which are available through the github are used for model validation in order to track performance and enable early stopping.

3.4 Models

Due to the mentioned difficulty of finding adequate data, my experiments will focus on Spanglish-English Language Normalization. Therefore, all selected teacher models have the ability to translate from Spanish to English at the very least. Models were selected based on their ability to translate these languages, as well as the number of supported languages in order to obtain a good representation of different language varieties. Wherever possible, the models will also have capabilities in English-English cleaning and paraphrasing. This will make tokenization of the input a lot more sensible and possibly boost performance considerably. The models, their supported languages and language groups, as well as their LinCE benchmark scores for the code-mixed test data are listed in table 3.3.

The teacher models are listed here, as the names tend to be very long, and are afterwards named in accordance with this list:

1. Model 1: Tatoeba-MT-models/cat+oci+spa-eng/ opusTCv20210807+bt_transformer-big_2022-03-13
2. Model 2: Tatoeba-MT-models/mul-mul/ opusTCv20230926+bt+jhubc_transformer-big_2024-08-17
3. Model 3: Tatoeba-MT-models/deu+eng+fra+por+spa-mul/ opusTCv20230926max50+bt+jhubc_transformer-big_2024-05-30

The model names include the version of the Tatoeba Challenge set the models have been trained on (e.g. TCv20230926 is the Tatoeba CHallenge set from 26.09.2023), whether or not the training data was limited to a maximum of 50M sentences per language pair (max50), whether or not the training data included backtranslations (bt) which are based on Wikipedia dumps and can be found on the Tatoeba GitHub page under https://github.com/Helsinki-NLP/Tatoeba-Challenge/blob/master/data/Backtranslations.md, and whether or not the training data included bible translations (jhubc - JHU Bible Corpus).

Model	Supported Languages	LinCE Performance
Model 1	cat+oci+spa-eng	40.18
Model 2	mul-mul	42.92
Model 3	deu+eng+fra+por+spa-mul	43.62

Table 3.3: Teacher models

Since the teacher models are all based on the transformer-big architecture introduced in Vaswani et al. (2017), effective model compression of student and baseline models into two smaller versions is possible in order to investigate the effects of Knowledge Distillation on increasingly smaller models. One of these student and baseline versions corresponds to the architecture of the base transformer proposed in Vaswani et al. (2017) and one is of the transformer-tiny architecture proposed in the Bergamot project found under https://github.com/browsermt. The detailed architecture and its differences are listed in Table 3.4. Each version of student and baseline models will have 5 different models which are trained with the same training data but different seeds in order to obtain a wider range of possible model performances to draw general conclusions for the models' level of compression. In total, therefore, I have trained 50 models of different sizes: 5 transformer-sized students for each of the three teachers, 5 transformer-sized baseline models for each of the two datset versions, as well as 5 transformer-tiny-sized models for each teacher and for each baseline.

This combination of models allows to not only draw general conclusions about model compression and the effect of Knowledge Distillation when compared to their bigger teacher models, both for compression from transformer-big to transformer architecture and from transformer-big to transformer-tiny architecture, but also allow for statistical analysis for statistical significance when compared to baseline models which have been trained from scratch. Practically, training the baseline models is achieved by using the same pipeline the student models are trained on, but instead of using the dataset that has been translated by the teacher models, the baseline models are using the datasets after cleaning and before the translation step takes place. This ensures that the training

procedure is the same and the only differences between student and baseline models is the fact that the student models were trained using Knowledge Distillation.

	Transformer-big	Transformer	Transformer-tiny
Number of Layers	6	6	6 encoder 2 decoder
Word Vector Dim	1024	512	256
Feed-Forward Dimension	4096	2048	1536
Number of Heads	16	8	8

Table 3.4: Model sizes

My experiments are divided into two phases.

Phase 1 focuses on training 5 baseline models for each student transformer architecture, which are trained using the pipeline. The models are trained with different seeds to ensure variablity in model performance and investigate model performance of different models of the same architecture in order to gain an estimate what to expect in an unrelated but similar training scenario for Language Normalization tasks for Spanglish to English. Performance on the Code-mixed test set is recorded for comparison with the student models as well as for comparison between transformer and transformer-tiny models of the same dataset. This means I have 5 baseline models for the 2021 Tatoeba dataset of transformer size architecture and 5 for transformer-tiny architecture, and the same applies to the 2023 Tatoeba dataset.

Phase 2 includes training student models for each teacher model. 5 student models are trained for both the transformer and transformer-tiny architecture for each of the three teacher models. All of the models are trained with different seeds to ensure variety in final trained parameter setups. Performances on the code-mixed test set are recorded, analyzed, and compared to the baseline models' performance as well as to teacher model performance, as well as to students of the same teacher model of different size.

3.5 Analysis

The analysis of model performances consists of comparing the performance of teacher models to their average student model performance as well as to the corresponding average baseline model performance. Performances are recorded as BLEU scores. As the teacher models only have one score, a performance is considered significant based on two setups: If teacher model performance is outside of the range of the mean performance of a model set including the standard deviation, the can be seen as mildly statistically significant since there does seem to be a statistical significance between the performance of teacher model and student or baseline model group. However, if the teacher's BLEU score is

higher or lower than the highest or lowest score of a model set, including the standard deviation of the student or baseline model group, the result is classified as statistically significant since there seems to be a clear difference between teacher model and student or baseline group performance. This strict measuring allows for a clear identification of the effectiveness of Knowledge Distillation for compressed models and also what effect the teacher model's performance has on student models compared to baseline models.

Statistical analysis is performed on the performances of each pair of student and baseline models of the same architecture and training data. As they were trained with the same dataset, the only difference is the fast that one is trained using Knowledge Distillation and one is trained from scratch. This means that the transformer students of teacher model 1 are compared to transformer baseline 2021 version models and transformer students of teacher models 2 and 3 are compared to the transformer baseline 2023 version models. The same applies to the transformer-tiny models. As the sample sets of each model group are rather small, with only 5 samples each, the results have to be interpreted with caution; however, they do give a good approximation of the measurable effect of Knowledge Distillation on model performance. For statistical analysis, the Mann-Whitney U-test is performed as it works rather well for small sample groups and does not assume a normal distribution.

Finally, I also compare the transformer performance against the transformer-tiny performance for each student and baseline set to see if continuous compression of the models changes performance. Comparisons will therefore be also performed for the transformer and transformer-tiny students of teacher model 1, transformer and transformer-tiny students of teacher model 2, transformer and transformer-tiny students of teacher model 3, transformer and transformer-tiny baseline 2021 models, and transformer and transformer-tiny baseline 2023 models.

Moreover, I compare model predictions of specific sentences that include special features, such as colloquial expressions and expressions specific to Mexican Spanish, that models might encounter if used in Language Normalization tasks for natural language. As the gold sentences are sadly not available, this is rather limited and focuses on specific phrases that are manually selected. Manual translations for the sentences were created by a native speaker in order to make an investigation of translation quality possible. For selecting sentences, several approaches were tested:

1. Using the cosine similarity measure for the predictions of each model in a model group in order to obtain an average cosine similarity across all models for all sentences of the test set. This allows for a ranking of the predictions. While this approach mostly takes into account how uniform a normalization of a sentence is,

it can be used as an indicator of how easy or hard a given sentence is to translate consistently for a model and therefore serves as a starting point for more detailed analysis. Choosing then the sentences that models commonly struggle with as well as sentences that models commonly do not struggle with might give insights into how Knowledge Distillation supports normalization when dealing with code-mixed input.

2. MarianNMT includes a scorer feature which can be used to score the translation quality of a sentence by calculating the cross-entropy between source and target sentence. This is used during the pipeline in order to filter out poor translations from the teacher model. Using this scorer for the code-mixed test data, a measure can be obtained for how good the translations are, even though these scores do not take the gold sentences into account and therefore have to be interpreted with care. Using these scores I am able to find sentences which seem to be easy or harder to translate based on how well they score across models.

3. - Quality Estimation Specia et al. (2018) is a useful tool to score translation quality of a sentence, even without access to gold sentences. As this fits my use case perfectly, I try using Unbabel's COMET QE (found here: https://github.com/Unbabel/COMET for scoring normalizations.

4. Using a simple scoring algorithm, the original test set sentences were scored based on how high the ratio of alphanumeric tokens was. This approach aimed to eliminate most of the social media induced noise, such as links, hashtags, and emojis, so that the sentences were as clean as possible and if possible only included code-mixing as "noise". The scores were calculated using a simple approach of counting the non-alphanumeric tokens in the sentence, subtracting this from the number of tokens in the sentence, and weighing it by the length of the sentence, also represented by the number of tokens. This gives a score between 1 and 0 depending on how many alphanumeric tokens are present in the sentence. This approach minimizes the amount of social media noise in the highest-scoring sentences. Sentences for detailed analysis can then be manually selected.

Out of these three approaches, I chose the fourth one to continue with. The issue with the cosine similarity approach is, that while it gives a good overview over how the reduction of multimodality is still present, it does not indicate actual translation quality. A lot of higher scored sentences were either not translated or very simple translations where the sentence consists of only a few words, often times with hashtags and links included. The issue with the second and third approaches was that the scores did seem

to have the same issue; sentences that were, in general, already easy due to the amount of special tokens in them scored quite high.

The fourth approach, however, presented me with a selection of sentences that were well representative of the goal of my thesis as they were almost entirely clean language with code-mixing and no hashtags, links or emojis. Since my goal is to inspect how translation models normalize code-mixed and code-switched text, not how they deal with highly noisy text, although this is obviously present in the BLEU scores for the entire test set, the sentences selected through their cleanliness score allow for investigation of how models deal with code-mixed input for Language Normalization and how they deal with special challenges that might occur in an industrial or professional use case. For my detailed analysis, I chose 6 of the highest ranked sentences according to the cleanliness scores. These sentences are sentences 8 and 1110, which have a higher ratio of Spanish than English, 21 and 4023, which have about equal amounts of both languages, and 1216 and 5047 which have a higher ratio of English than Spanish. These sentences have barely other noise than the code-mixing present in the dataset and moreover have specific challenges that might make the work for translation models harder, such as colloquial expressions or country-specific terms.

I accept the premise that knowledge distillation makes models more or less robust in dealing with code-mixing, if the majority of student models outperform their baseline counterparts and their teacher models for the code-mixed test set. The sentence analysis results are also taken into account, mainly to highlight models' abilities to deal with special cases.

Chapter 4

Results

Table 4.1 shows the BLEU scores for student and baseline models for the code-mixed test set as well as illustrates the results of the statistical analysis. The performance scores for tasks that the models are trained for can be found in appendix A and appendix B. Illustrated graphs for model performances with their pvalues included are shown in Appendix A and Appendix B. The graphs' y-axis are based on the matplotlib log scale to illustrate differences in performance better. As this makes the steps on the y-axis more difficult to read, the individual model performances are added to each bin. Moreover, while the results and discussion chapter takes into account the student and baseline models' performances and comparisons for the two tasks the models were trained for, Spanish-English translation and the English-English mixed task, the main indicator of the confirmation or rejection of my research hypotheses depends on the performance in the Language Normalization task. While the other two tasks are included for completeness, they do not play into the final conclusion of my thesis in the same way that the Language Normalization task does.

For teacher model 1 students, the individual performance scores as well as the p-value for student and baseline models for code-mixing, Spanish-to-English translation, and English-to-English task can be found illustrated in the Appendix in figure A.3, A.4, A.5, B.3, B.4, and B.5.

For the transformer-sized models, student models of teacher model 1 show a significantly better performance for code-mixing than the teacher model, while the baseline model does only show a mild statistical significance. However, there is no statistical significance between student and baseline models. This is also true for the Spanish-to-English task, although in this case, both students and baseline models are significantly worse than the teacher model. Interestingly, student models show significantly better performance than the teacher model and the baseline model for the English-to-English

task. Additionally, the baseline models do not show a significantly different performance for this task compared to the teacher model.

For transformer-tiny students of teacher model 1, the situation is different. Student models are significantly better than baseline models in the case of code-mixed Language Normalization, however, only mildly significantly better than the teacher model. Baseline models are worse than the teacher model, but not significantly. For Spanish-to-English translation, student models are significantly better than baseline models, but both student and baseline models are significantly worse than the teacher model. It seems that while noise-handling seems to have gotten better, actual translation has gotten worse with further compression. Student models show significantly worse performance for the English-to-English task when compared to baseline models, even though the p-value is close to 0.05. At the same time, the teacher model shows significantly worse performance compared to both student and baseline models. This particular performance, however, might also be influenced by the skewed ratio of the training data, as it has a much higher amount of Spanish-to-English sentence pairs than English-to-English.

When comparing transformer and transformer-tiny models, the transformer models were significantly better than the transformer-tiny models in Language Normalization and Spanish-to-English translation, but not significantly better in the English-to-English task, showing a clear decrease in performance when compressing models further.

Metric	Transformer	Transformer-tiny
Teacher Model 1 BLEU	40.18	
Student Mean	41.966	41.102*
Student Standard Deviation	0.3255	0.9395
Teacher Model 2 BLEU	42.92	
Student Mean	52.664**	50.836**
Student Standard Deviation	0.6206	1.0957
Teacher Model 3 BLEU	43.62	
Student Mean	53.234**	52.794**
Student Standard Deviation	0.3563	0.2611
Baseline 2021 Mean	41.884	39.54
Baseline 2021 Standard Deviation	0.9395	0.5316
Baseline 2023 Mean	39.18	40.084
Baseline 2023 Standard Deviation	0.8114	0.6768

Table 4.1: Student and Baseline Model Performances for the Spanglish-English Language Normalization task as BLEU scores. Statistical significance with $p<0.05$ for student-baseline comparisons is indicated by a * behind the BLEU score mean values that are significantly better, while statistical significance with $p<0.01$ is indicated with **

Teacher model 2 students show a similar situation when compared to their teacher

and baseline models. The performance scores as well as the p-value for each student and baseline model for code-mixing as well as Spanish-to-English translation and English-to-English cleaning can be found illustrated in the Appendix in figure A.6, A.7, A.8, B.6, B.7, and B.8.

Transformer-sized models for teacher model 2 show statistical significance for student and baseline model groups for the code-mixed test set. Additionally, student models are significantly better than the teacher model, and baseline models are significantly worse than the teacher model. However, student models are significantly worse than baseline models in Spanish-to-English translation. In comparison to their teacher, student models show mildly significantly better performance, while the baseline models are significantly better than the teacher model. For the English-to-English task, however, student models again show significantly better performance than the baseline models, with the teacher model being mildly significantly better than the student models. The teacher model is, however, significantly better than the baseline models, showing that student models tend to mimic their teachers on tasks they are trained for.

The transformer-tiny models for teacher model 2 outperform the baseline models significantly in the code-mixed test set and the English-to-English task, while showing significantly worse performance for the Spanish-to-English translation task. The tiny student models for teacher model 2 are also the first models to perform better for the Language Normalization task than for one of the two Tatoeba test set tasks, in this case the Spanish-to-English translation. Student models 3, 4, and 5 performed better than the Spanish-to-English translation, to be exact, and lead to a slightly higher mean performance for Language Normalization for the student model group. Overall, student performance in Spanish-to-English translation is, however, still significantly worse than baseline models. Compared to the teacher model, the student models outperform the teacher significantly for the code-mixed test set, showing the effectiveness already visible for teacher model 1, that even very small models can benefit from Knowledge Distillation. For the Tatoeba test set tasks, the student model does not manage to significantly outperform the teacher model, showing mildly significant improvement for the Spanish-to-English task and significantly worse performance for the English-to-English task. The baseline models show significantly worse performance for the code-mixed test set as well as mildly significantly worse performance for the English-to-English task. However, baseline models are significantly better than the teacher model for the Spanish-to-English translation task, which again shows that if a teacher model is not performing well in certain tasks, their students might be limited by their teacher's performance.

In comparison with transformer-tiny students, the transformer-sized students of teacher model 2 outperform the tiny models in the code-mixed test set significantly, while only

showing slightly better performance for Spanish-to-English translation and a slight decrease in the English-to-English task.

Teacher model 3 student and baseline models again exhibit a similar situation with some differences compared to the other teachers and their models. The performance scores as well as the p-value for student and baseline models for code-mixing as well as Spanish-to-English translation and English-to-English cleaning can be found illustrated in the Appendix in figure A.9, A.10, A.11, B.9, B.10, and B.11.

Students of teacher model 3 of transformer size show the same significance as their baseline models as teacher 2 students, meaning a significantly better performance for the code-mixed test set as well as for the English-to-English task, and significantly worse performance for Spanish-to-English translation when compared to their baseline counterpart. Student models are also similarly performing regarding to their teacher model, showing significant improvement in the code-mixed test set and mildly significant improvement for the Spanish-to-English translation. However, while the students in teacher model 2 only show a slight decrease in English-to-English performance while for the students of teacher model 3, the decrease is significant. This might be an effect of the number of involved languages for the teacher models, as teacher 2 is able to work with many more languages as source languages which might enable the model to leverage overlapping information between related languages more effectively. The performance relation between teacher model 3 and baseline models is the same as for teacher model 2, with the teacher model being significantly better than the baseline models for the code-mixed test set, significantly worse for the Spanish-to-English translation, and significantly better for the English-to-English task.

The situation is different for the transformer-tiny students for teacher model 3, however. While the student models are still significantly better than the baseline models and the teacher model for the code-mixed test set, the students are not significantly worse than the baseline models for the Spanish-to-English translation, but significantly worse than the teacher model, which was not the case for the transformer-sized students. For the English-to-English task, the student models are significantly better than their baseline models and mildly significantly worse than the teacher model. The baseline models show significantly worse performance than the teacher for the code-mixed test set and mildly significantly worse performance the English-to-English task. However, for the Spanish-to-English translation, the baseline models were not significantly better than the teacher model.

When comparing the transformer-sized models to transformer-tiny-sized models for teacher 3, performance is not significantly decreased for the tiny models for the code-

mixed test set. This is the only time this is happening for student models, as both teacher 1 and 2 student models had a significant decrease in code-mixed test set scoring with further compression.

The performances for the code-mixed test set for teacher 2 and 3 transformer models have to be seen with care, however, as transformer-sized student models 5 for teacher 3 predicted an empty string for sentences 2744, 4893, and 6051, which caused the website for the evaluation of the predictions to throw an error during evaluation since the empty lines were not taken into account. Moreover, teacher model 2's transformer student 5 also predicted an empty string for sentence 4893. I therefore edited the affected lines to a single full stop. I chose this because it does not impact the sentence prediction too much, but it has to be taken into account that the prediction score for student model 5 for teachers 2 and 3 might be slightly lower than reported in 4.1.

Sentence 4893 is

> 🥭 + 🍸 + 🐓+ <user> + sorteos = noche de gozadera . RSVP #VerizonAccess : https://t.co/CzxytSdzCk https://t.co/CNWy4WR9HM

which does not give a lot of context to translate, even less so a proper grammar for a full sentence. All other student models of teacher model 2 predicted a sentence, although there are a lot of unknown tokens, and oftentimes the few words of the sentence are not correctly translated. However, the teacher model prediction is a proper sentence that translates the available words correctly. Out of all the transformer-sized baseline models, only one created a sensible translation for sentence 4893, while all others produced nonsense translations. Out of other teacher and student models, teacher 2 produced a nonsense translation, while 3 of its student models still produced somewhat sensible translations. Teacher 1 produced a partial translation, and only 3 of its student models produced a partial translation as well; the rest are nonsense translations. Given the nature of the sentence, it makes sense that student models are struggling with producing a translation, especially considering that even the bigger teacher models are struggling with sentence 4893.

The statistical analysis of the baseline model sets for both data versions shows that both versions exhibit the same performance relations for the Spanish-to-English and the English-to-English task, being that the transformer models are significantly better than the tiny models for Spanish-English and significantly worse for English-English. Interestingly, however, while for the Language Normalization task the baseline 2021 version's smaller models showed significantly worse performance, the baseline 2023 version models showed slightly increased performance, though it is not statistically significant.

While performance scores seem to show benefits of Knowledge Distillation for code-mixed Language Normalization tasks, there are also benefits to training and inference times. The inference times for individual models shown in table C.1 show that sometimes the first model takes a lot longer to complete the code-mixed test set translations. This is caused by the Slurm job. Since the translation jobs are submitted to the university server in combined Slurm jobs, the first models in these combined jobs tend to take a bit longer since the environment has to be set up and the needed modules need to be activated. This causes the first model for baseline 2021 transformer, baseline 2023 transformer, teacher model 3 transformer, baseline 2021 transformer-tiny, teacher model 1 transformer-tiny, as well as the 5th model for teacher model 2 transformer, since this was translated separately due to file issues, to have a longer translation time than the others. Because of this, in order to determine the average translation times, instead of the arithmetic mean, I am using the median of each measuring group for the shown translation times in 4.2. This is not as precise as the arithmetic mean, however, the outliers will not have as big an effect as they would have when using the mean.

The shown inference speeds show a very significant increase for student models compared to teacher models and a further increase for the even smaller models, with all student models performing similarly quickly compared to baseline models. Student models for teacher model 1 of transformer size are slightly quicker at inference than their baseline model counterparts while the transformer-tiny models are slightly slower. Student models of teacher model 2 of transformer size are slightly faster than their baseline counterparts while being showing a similar improvement to their baseline models when compared to their teacher inference time. For teacher model 3 students, the situation is the same as for teacher model 2 students, which has interesting implications regarding a regularlity as in both cases the transformer students are 0.3% quicker than the transformer-sized baseline models but a very similar improvement for transformer-tiny-sized models. Teacher model 1 students likely have a different situation due to the different training data sets. However, this difference would need further investigation with identical training sets to be confirmed.

The training times shown in 4.3, C.2, and C.3 are taken from Marian-created training logs, ranging from the "training started" log messages to either "Training completed" or until the last saved checkpoint. Since the teacher models had pauses while training before resuming later on, the training times are drastically increased from their actual time spent training. For this reason, teacher models' times are only taken into account from each "Training started" log message until the last training update before the new training run resumed and summed up at the end. This way, a more accurate estimate of

Model version	Model size	Inference speed	Duration decrease
Model 1	Teacher	7.0 sent/sec	-
	Student-Transformer	286.3 sent/sec	97.5%
	Student-Transformer-Tiny	670.1 sent/sec	98.9%
Model 2	Teacher	5.4 sent/sec	-
	Student-Transformer	310.3 sent/sec	98.3%
	Student-Transformer-Tiny	659.2 sent/sec	99.2%
Model 3	Teacher	6.4 sent/sec	-
	Student-Transformer	303.5 sent/sec	97.9%
	Student-Transformer-Tiny	708.8 sent/sec	99.1%
Baseline 2021	Transformer	262.3 sent/sec	97.3%
	Transformer-Tiny	701.9 sent/sec	99%
Baseline 2023	Transformer	269.4 sent/sec	98% (Teacher 2) 97.6% (Teacher 3)
	Transformer-Tiny	721.4 sent/sec	99.2% (Teacher 2) 99.1% (Teacher 3)

Table 4.2: Translation speeds in sentences per second

teacher training times can be formed and is presented in table 4.3. However, it has to be mentioned that a direct comparision between teacher and student training times is not possible due to the large differences in the used training sets as teacher models included not only more languages in their training data, but also backtranslations and bible translation texts. Nevertheless, the information is useful when it comes to the differences in training times between students to highlight possible causes for these differences across student model groups, a lot of which are likely caused by the different trainig sets and the skewed ratio of languages in the 2021 set. Direct comparison is possible for student and baseline models as the baseline training sets are the same with the differences being only the filtered sentences after translation for the student models. This difference in training set sicze likely has an effect on training time as well, however, given the differences listed in 4.3 it is unlikely that the only cause is the difference in training set size.

While training times decrease with further model compression generally, baseline 2021 models are an exception with an increase in average training time when training transformer-tiny models instead of transformer models of 1 hour and 11 minutes, which might be because of the skewed data ratio in the training data which, coupled with the reduced capacities for the student models of smaller size, account for the delayed convergence of the models.

Teacher models show very long training times, as expected of models of the size and with the teaining data available. However, as mentioned, these training times will not play into the actual comparison of reduction of training times and are merely for illustrative purposes. Training time for students is lower than for baseline models in all cases, show-

ing benefits of using Knowledge Distillation for training smaller yet still capable models. When compared to their respective baseline model groups, teacher 1's students of transformer size show a reduction fo training time of 25.1% with their transformer-tiny versions even increasing this difference to 33.8% compared to the baseline 2021 transformer-tiny models. Teacher 2's student models of transformer and transformer-tiny size have a reduction of training time of 51.4% and 17.3% respectively in comparison to their baseline 2023 transformer and transformer-tiny models. Here it shows again that the smaller capacity of smaller models might be not enough to encode a large amount of information from many languages efficiently which leads to longer training times. Teacher model 3's students of transformer size show an even higher reduction in training time compared to their baseline 2023 counterparts with 72.3% faster training. The transformer-tiny students of teacher model 3 still show a reduction of training time of 23.7% which, given the pattern for teacher model 2's students, seems to be again linked to the number of languages the teacher model supports and the reduced capacity of student models of smaller size. It is interesting, however, that students of teacher model 1 show the lowest reduction of training time for transformer size, however, make up for that by having the highest reduction of training time for transformer-tiny models. The difference for transformer-sized models might be linked again to the skewed dataset for the 2021 version, while the strong reduction of training time for transformer-tiny students in comparison to the other student models might again be linked to the reduced capacity of smaller models which leads the models having an advantage thanks to their teacher model only supporting four languages overall.

Model version	Model size	Training Duration
Model 1	Teacher	455h 19min 18sec
	Student-Transformer	17h 23min 16sec
	Student-Transformer-Tiny	16h 09min 12sec
Model 2	Teacher	1,002h 40min 24sec
	Student-Transformer	18h 08min 40sec
	Student-Transformer-Tiny	16h 06min 20sec
Model 3	Teacher	372h 32min 17sec
	Student-Transformer	18h 39min 19sec
	Student-Transformer-Tiny	14h 51min 58sec
Baseline 2021	Transformer	23h 13min 09sec
	Transformer-Tiny	24h 24min 13sec
Baseline 2023	Transformer	37h 19min 39sec
	Transformer-Tiny	19h 28min 21sec

Table 4.3: Average student and baseline model set training durations and accumulated Teacher training speeds

The following paragraphs focus on the presentation of the results from the detailed sentence analysis for six selected sentences from the code-mixed test set. As mentioned in section 3.5, the sentences were chosen to investigate how models perform when faced with different challenges that might occur during coimmercial use of a Machine Translation model for Language Normalization tasks.

Tables 4.4 and 4.5 show the original sentence as well as normalizations by a native speaker, the teacher, student, and baseline models for the first sentence to be analyzed, sentence 8 of the code-mixed test set.

The sentence is mostly Spanish with one English word, ”paper”. The sentence also includes some colloquial expressions, which might be a challenge for the models. Teacher model 1 adds a space at the beginning of the sentence for all sentences tested, which will be disregarded for the rest of the sentences as this is a regular occurrence. The sentence is itself well translated, but ”con cojones”, while pragmatically and even semantically correct, is slightly off in its literal presentation. All student models of transformer size for teacher model 1 translate the sentence like the teacher model, except for student model 1, which replaces ”balls” with ”lameons,” which might be a generation error, or an erroneous generation of ”lemons”, which would make the translation slightly different in severity of its vulgarity. Overall, however, the sentence structure and semantics are preserved and, in most cases, identical to the teacher model, meaning the literal presentation of ”con cojones” is not shown. None of the translations of the transformer-tiny students of teacher 1 are error-free; every model has at least one type of error. Some have wrong word choices or some grammatical issues. All of them still preserve the pragmatics of the sentence; however, models 1 and 3 seem to struggle with ”con cojones” as they translate it to something completely different, and even though the other models are not close in their semantics for the sentence, these two models are still more off. Moreover, model 5 has a wrong translation for ”Comiendo”, which has not happened for any other student model or their teacher.

Teacher model 2 adds a pronoun at the beginning as well as not catching the exact way to translate ”con cojones” which is translated as ”coons” instead of ”ballsy”, which could either be a generation error, or a translation into a colloquial term ”coon” (as found in urban dictionary under https://www.urbandictionary.com/define.php?term=coon%29&page=5). Otherwise, the translation is okay. However, out of teacher model 2’s transformer-sized students only one student model (model 3) has a relatively close to correct translation without clear errors in generation or translation. ”mierda” is translated in two different ways, one of which is grammatically and lexically wrong, however, apparently, the student models struggle with the emphasizing expression ”con cojones” which the teacher model struggles with as well. This sentence shows well Knowl-

Model	Sentence
Original	Comiendo mierda con cojones para hacer este paper
Native Translation	Eating shit in a ballsy way/ballsily to do this paper
Teacher 1	Eating shit with balls to make this paper
Students - Transformer	Eating shit with lameons to make this paper Eating shit with balls to make this paper Eating shit with balls to make this paper Eating shit with balls to make this paper Eating shit with balls to make this paper
Students - Tiny	Eating shit with lams to make this paper Eating shit with the fuck to make this paper Eating shit with lanes to make this paper Eating shit with the fuck to make this paper Suffoning fuck with fucks to make this paper
Teacher 2	I'm eating shit with coons to make this paper
Students - Transformer	Eating shit with beans to make this paper Eating fucking with the fucks to make this paper Eating shit with fucks to make this paper Eating fucking with chickens to make this paper Eating shit with chickens to make this paper
Students - Tiny	Eating shit with coats to make this paper Eating fucking with fucks to make this paper Eat shit with fucks to do this paper Eating shit with fucks to make this paper Eating fucking with coats to do this paper
Teacher 3	Eating fucks to make this paper
Students - Transformer	Eating shit with fucks to make this paper Eating shit with fucks to make this paper Eating shit with fucks to make this paper I eat shit with fucks to make this paper Eating shit with fucks to make this paper
Students - Tiny	Eating shit with fucks to make this paper Eating shit with fucks to make this paper Eating shit with fucks to make this paper Eating shit with fucks to make this paper Eating shit with fucks to make this paper

Table 4.4: Sentence 8 - Student and Teacher Normalizations

Model	Sentence
Original	Comiendo mierda con cojones para hacer este paper
Native Translation	Eating shit in a ballsy way/ballsily to do this paper
Baseline 2021 - Transformer	Eating shit with coons to make this paper Eating shit with cojones to make this paper Eating shit with cojones to make this paper Eating shit with fucks to make this paper Eating shit with balls to make this paper
Baseline 2021 - Tiny	Eating shit with fucks to make this paper Eating shit with the fuck to do this paper Eating shit with the fuck to do this paper Eating shit with fuck to make this paper Eating shit with the fuck to make this paper
Baseline 2023 - Transformer	I'm eating shit with fucking balls to do this paper. Eating fuck with balls to make this paper Fucking fuck with balls to make this paper Eating Fuck With Them to Make This Paper Eating shit with balls to do this paper
Baseline 2023 - Tiny	Eating shit with pussocks to make this paper Eating shit with pussy to make this paper Eating shit with balls to make this paper Eating shit with cocks to make this paper Eating shit with balls to make this paper

Table 4.5: Sentence 8 - Baseline Normalizations

edge Distillation's possible issues of inheriting issues the teacher model is exhibiting already. Additionally, some models choose to interpret "cojones" as "eggs" and translated it as "chickens", which is an obvious lexical error. The tiny version of student models for teacher model 2 has two correct translations, two models have a grammatical error, "fucking" instead of "shit", and two models struggled with "cojones" and generated "coats" instead, which has no semantic or pragmatic connection to the original sentence. The overall pragmatics of the original sentence seems to be preserved for the models that did not generate "coats".

Teacher model 3 shortens its translation from "mierda con cojones" to a single "fucks", which can be attributed to the fact that the expression can be seen as a single expression based on "mierda" and being emphasized by "con cojones". "shit" would be a closer translation to the literal version, but as "fucks" carries similar pragmatics in this context it is an acceptable variation. All translations of teacher model 3's transformer-sized students are pragmatically correct, although they are missing the literal and semantic meanings of "con cojones". One model changes the beginning of the sentence, but the pragmatics remain the same. No model seems to struggle with the source sentence as a whole; however, student models seem to normalize the sentences very uniformly. All transformer-tiny students of teacher model 3 translate the sentence exactly the same, and all translations are pragmatically correct. No model seems to struggle with the sentence, even though they are a lot smaller than the first students, and even though they do not take the more literal meaning of "con cojones" into account.

Only two baseline 2021 transformer-sized models have a pragmatically correct translation of sentence 8, one of them is a more literal translation, translating "cojones" as "balls". The other models seem to struggle with "cojones", one of them again translating it to "coons", two of them not translating the word at all. Compared to the students of teacher 1, there are more issues with these translations: Models seem to be struggling with "con cojones" more than the student models, as there is no literal translation for it. All translations of the baseline 2021 transformer-tiny models are pragmatically correct. However, there are some grammatical issues with three models' translations, where "fucks" is not in the correct form, instead being "the fuck". Compared to the bigger students, however, these translations seem to be qualitatively better and more literal.

Three translations of the baseline 2023 transformer-sized models are pragmatically correct, with varying word choices and degrees of literal correctness. Two translations have wrong word choices, one for the beginning of the sentence, one for "cojones". All translations of the baseline 2023 transformer-tiny models are pragmatically correct for the entire sentence, translating "cojones" as emphasizers for "mierda" and only varying in word choice for this purpose. There are no grammatical errors in the translations,

showing that the smaller models' translation power is still present despite being trained from scratch. It is interesting, however, that these translations seem to be more uniform than the ones of baseline 2021 of the same size, which seems to be the case because they all have the appropriate forms for their "con cojones" translation, while baseline 2021 transformer-tiny models have some grammatical errors there. The only model set that normalizes the sentence more uniformly than these models is the transformer-tiny student set of teacher model 3.

Tables 4.6 to 4.10 show the second sentence for analysis, sentence 1110. The tables include the original sentence as well as native and machine normalizations.

The sentence has a higher Spanish than English ratio, testing models' capacity of translating more than paraphrasing/retaining info. The sentence contains a named entity "Ramires de Arellano" which most likely refers to the "Residence Ramirez De Arellano", meaning models' capabilities of named entity recognition play an important role in these translations.

Teacher model 1's normalization is almost correct, shown in table 4.6, and the named entity is preserved. The only issue is the fact that the active part of touching is switched to the writer and not the other person, as well as the wrong preposition for the named entity. The fact that the teacher model misattributed the verb "touch" already indicates that student models could struggle with this part as well. All transformer-sized student models for teacher model 1 preserve the named entity; however, no model has the correct preposition "of" as two had "in" and three had "on". Moreover, only one model preserves the semantics of the original sentence, one creates some grammatical errors in the beginning of the sentence, and three models translate the beginning into "let me touch" instead of "If a slow person touches me". This might be because of colloquial language and missing context. None of the transformer-tiny student models produce fully correct translations; however, model 2 gets close, if not for a missing "by". Two models again translate the beginning of the sentence as "let me" and model 1 produces a grammatically correct sentence that carries similar pragmatic information as the original sentence, but fails to incorporate the "touch" or "toque" correctly, instead generating "getting a slow person in front" which is not equivalent to the original sentence. Additionally, models 2 and 3 break up the named entity, indicating that smaller models might have trouble with this task.

Teacher model 2's normalization, shown in table 4.7 is not grammatically correct, but the named entity is preserved. The translation seems to have switched the active person of the "touch" again, just like teacher model 1. Except for student model 2, all transformer-sized student models of teacher model 2 translate the sentence with "let me" at the beginning. Model 2 is the only one to produce a completely correct translation,

Model	Sentence
Original	Que me toque una persona lenta al frente en la Ramírez de Arellano está en mis top 3 worst things in life
Native Translation	If a slow person touches me in front of the Ramirey de Arellano it would be one of my top 3 worst things in life
Teacher 1	That I touch a slow person in front in the Ramírez de Arellano is in my top 3 worst things in life
Students - Transformer	Let me touch a slow person in front on the Ramírez de Arellano is in my top 3 worst things in life Let me touch a slow person in front on the Ramirez de Arellano is in my top 3 worst things in life Let me have a slow person at the front on the Ramirez de Arellano is in my top 3 worst things in life That touches me a slow person in front in the Ramirez de Arellano is in my top 3 worst things in life That I'm touching a slow person in front in the Ramírez de Arellano is in my top 3 worst things in life
Students - Tiny	Getting a slow person in front at the Ramirez de Arellano is in my top 3 worst things in life That I'm touched a slow person in front at Arellano's Ramirez is in my top 3 worst things in life Let me touch a slow person at the front in Arellano's Ramirez is in my top 3 worst things in life Let me touch a slow person in front in the Ramírez of Arellano is in my top 3 worst things in life I'm touching a slow person at the Ramírez de Arellano is in my top 3 worst things in life

Table 4.6: Sentence 1110 - Student and Teacher Normalizations - Teacher Model 1

which is interesting considering that the teacher model did not manage to do this and seems to indicate possible improvements to not only noise handling but also adaptability for very capable teacher models when using Knowledge Distillation. Sadly, none of the transformer-tiny student models for teacher model 2 produce accurate normalizations, though model 5 gets close, only adding an additional "me" in front of the sentence. Grammatically, model 5's "at the Ramirez" is not correct; however, this can be attributed to the colloquial writing of the tweet. Model 2 again translates the beginning of the sentence as "Let me". All models except 3 and 4 keep the named entity intact, however. Models 3 and 4 changes the named entity "Ramirez de Arellano" to "Arellano Ramirez", breaking up the entity just like the tiny models of teacher model 1 which is further evidence that increasingly smaller models seem to struggle with keeping named entities in Language Normalization tasks.

The normalization of teacher model 3, shown in table 4.8 is correct except for a

Model	Sentence
Original	Que me toque una persona lenta al frente en la Ramírez de Arellano está en mis top 3 worst things in life
Native Translation	If a slow person touches me in front of the Ramirey de Arellano it would be one of my top 3 worst things in life
Teacher 2	To touch me a slow person in front in the Ramírez de Arellano is in my top 3 worst things in life
Students - Transformer	Let me touch a slow person in front in the Ramirez de Arellano is in my top 3 worst things in life That I touch a slow person in front of the Ramírez de Arellano is in my top 3 worst things in life Let me touch a slow person in front of the Ramírez de Arellano is in my top 3 worst things in life Let me touch a slow person in front of the Ramírez de Arellano is in my top 3 worst things in life Let me touch a slow person in front of the Ramírez de Arellano is in my top 3 worst things in life
Students - Tiny	That touches me a slow person in front of the Ramírez de Arellano is on my top 3 worst things in life Let me touch a slow person in the Ramírez de Arellano is in my top 3 worst things in life That touches me a person slowly in front of the Arellano Ramírez is on my top 3 worst things in life That touches me a slow person in front of the Arellano Ramirez is in my top 3 worst things in life To touch me a slow person in front at the Ramírez de Arellano is in my top 3 worst things in life

Table 4.7: Sentence 1110 - Student and Teacher Normalizations - Teacher Model 2

grammatical error of ”touch” instead of ”touches”. The named entity is preserved as well. This teacher model is the first to do so. None of the teacher’s transformer-sized student models produces a correct translation; model 5 comes close but breaks up the named entity, just like model 1, and model 3 changes it from ”de” to ”of”. This is the first time as well that the transformer-sized models break up the named entity. Models 1-4 also add the extra ”me” at the beginning of the sentence, most likely a faulty interpretation of the ”me” in ”Que me”. Only model 3 is able to use the correct preposition for the named entity. Transformer-tiny student model 4 produces a pragmatically correct normalization, but two models break up the named entity. Only two models have issues with the sentence beginning, however, showing an improvement over the transformer-size models.

None of the transformer-sized baseline 2021 models, shown in table 4.9 produce a correct normalization. Model 3 comes close, but it has a grammatical error with ”touch” instead of ”touches” and ”at the front in” instead of ”in front of”. All models keep

Model	Sentence
Original	Que me toque una persona lenta al frente en la Ramírez de Arellano está en mis top 3 worst things in life
Native Translation	If a slow person touches me in front of the Ramirey de Arellano it would be one of my top 3 worst things in life
Teacher 3	That a slow person touch me to the front in Ramírez de Arellano is in my top 3 worst things in life
Students - Transformer	That touch me a slow person in front in the Arellano Ramírez is in my top 3 worst things in life That touch me a slow person in the front in the Ramirez de Arellano is in my top 3 worst things in life Touching me a slow person in the Ramirez of Arellano is in my top 3 worst things in life That touch me a slow person in front in the Ramirez of Arellano is in my top 3 worst things in life That I touch a slow person in front in the Arellano Ramirez is in my top 3 worst things in life
Students - Tiny	That I touch a slow front person on the Arellano Ramírez is in my top 3 worst things in life Touching me a slow person in the front in the Ramírez de Arellano is in my top 3 worst things in life Touch me a slow person in the Ramírez de Arellano is in my top 3 worst things in life Touching a slow person in front in the Ramirez de Arellano is in my top 3 worst things in life Touching a slow person in the Arellano Ramirez is in my top 3 worst things in life

Table 4.8: Sentence 1110 - Student and Teacher Normalizations - Teacher Model 3

the named entity intact. Models 3 and 4 of the transformer-tiny group produce a correct normalization; model 2 gets close but leaves out the ”front” part. Overall, the transformer-tiny versions seem better than the transformer-sized ones, as they also do not contain very wrong words like ”playing”.

None of the baseline 2023 transformer-sized models, shown in table 4.10 produce an entirely correct normalization; model 4 comes close, but chooses the wrong prepositions for the person and the named entity. Model 3 breaks up the named entity, and model 1 translates the beginning of the sentence as ”I play”. Model 5 comes the closest to a correct translation, but the preposition for the residence is wrong. Baseline 2023 transformer-tiny-sized model 5 breaks up the named entity, only model 1 produces a correct translation, and two models include the ”play” translation. Overall, the normalization quality is worse than transformer-size models, even though there is one correct version, but the others being worse than the average normalization of the transformer-sized models means

Model	Sentence
Original	Que me toque una persona lenta al frente en la Ramírez de Arellano está en mis top 3 worst things in life
Native Translation	If a slow person touches me in front of the Ramirey de Arellano it would be one of my top 3 worst things in life
Baseline 2021 - Transformer	I'm touched by a slow person in front of the Ramirez of Arellano is in my top 3 worst things in life I'm playing a slow person in the front in the Ramirez de Arellano is in my top 3 worst things in life That I touch a slow person at the front in the Ramírez de Arellano is in my top 3 worst things in life Touch me a slow person in front of the Ramírez of Arellano is in my top 3 worst things in life That a slow person touch me at the front in the Ramírez of Arellano is in my top 3 worst things in life
Baseline 2021 - Tiny	To touch me a slow person in front at the Ramirez de Arellano is in my top 3 worst things in life That I touch a slow person in the Ramirez de Arellano is in my top 3 worst things in life That I touch a slow person in front of the Ramírez of Arellano is in my top 3 worst things in life That I touch a slow person in front of the Ramirez de Arellano is in my top 3 worst things in life I'm touched a slow person at the front in the Ramirez of Arellano is in my top 3 worst things in life

Table 4.9: Sentence 1110 - Baseline 2021 Normalizations

that further compression did not improve overall performance.

Tables 4.11 to 4.15 show the third sentence for analysis, sentence 21. The tables include the original sentence as well as native and machine normalizations.

The source sentence has a rather equal split between languages as well as some colloquial writing, such as "ke" instead of "que" and "oh" instead of "o", which might prove an issue for the models. Additionally, "peores" is spelled wrongly as "piores", which might pose additional issues.

The normalization of teacher model 1, shown in table 4.11, has significant grammatical issues as well as some colloquial terms that are not translated at all, for example "piores". Overall normalization quality is poor. The last three words of the source sentence are not translated for all transformer-sized student models of teacher 1, as the deviation from "o" with "oh" in front of "piores" seems to confuse the models, so that it is not recognized as a Spanish word to be translated. Models 2-4 pick up on the colloquial "ke" and translate it correctly as "that" as opposed to the teacher model. However, overall grammar of the normalizations is not good, as the translations are rather literal. Model 2 is slightly closer

Model	Sentence
Original	Que me toque una persona lenta al frente en la Ramírez de Arellano está en mis top 3 worst things in life
Native Translation	If a slow person touches me in front of the Ramirey de Arellano it would be one of my top 3 worst things in life
Baseline 2023 - Transformer	That I play a slow person in front of me in the Ramírez of Arellano is in my top 3 worst things in life That I touch a slow person in front of the Ramirez of Arellano is in my top 3 worst things in life I'm a slow front person in the Ramirez Arellano is in my top 3 worst things in life To touch a slow person to the front in the Ramírez de Arellano is in my top 3 worst things in life That I touch a slow person in front on the Ramírez of Arellano is in my top 3 worst things in life
Baseline 2023 - Tiny	That I touch a slow person in front of the Ramirez de Arellano is in my top 3 worst things in life I'm a slow person at the Ramirez of Arellano is in my top 3 worst things in life I'm playing a slow person in the Ramirez de Arellano is in my top 3 worst things in life I'm playing a slow person in front of the Ramirez of Arellano is in my top 3 worst things in life Let me touch a slow person in front in Arellano's Ramirez is in my top 3 worst things in life

Table 4.10: Sentence 1110 - Baseline 2023 Normalizations

to the native normalization than the teacher model. Student model 1 of transformer-tiny size of teacher model 1 wrongly translates the sentence beginning "I" into "And". Student model 1 also has a variation of "sus hijos" as "your children" instead of "their children", which is okay. However, student model 1 does not translate "igual" while all other models do. Overall normalization quality is not very different from transformer-size models. All models except student model 4 do not translate "ke" at all and just omit it. In the sense of identifying colloquial language, the transformer-size models seem to perform slightly better, as one model translates "ke" correctly. Models 2 and 4 attempt to translate "piores", but the translations are wrong, likely due to the difference in the preceding word "oh" instead of "o".

The Spanish part of the source sentence is almost fully translated by teacher model 2, shown in table 4.12. "ke" is translated correctly. The only mistake is that "oh" is left out completely, leaving the sentence grammatically incorrect. Translations for the transformer-sized student models of teacher model 2 are quite similar to each other, model 4 chooses "your children" instead of "their children". Student models 1 and 4 do not create

Model	Sentence
Original	I hate when people talk shit cuzs I drink si supieran ke sus hijos tan igual oh piores ha
Native Translation	I hate when people talk shit cause I drink if they knew that their kids are the same or worse ha
Teacher 1	I hate when people talk shit cuzs I drink if they knew ke their children so equal oh piores ha
Students - Transformer	I hate when people talk shit cuzs I drink if they knew ke their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew that their children so evenly knew oh piores has I hate when people talk shit cuzs I drink if they knew their children so equally oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew ke their kids so equal oh piores has
Students - Tiny	And hate when people talk shit cuzs I drink if they knew your kids so much oh piores ha I hate when people talk shit cuzs I drink if they knew her children so equal, oh sleeper has I hate when people talk shit cuzs I drink if they knew their children so equal oh piores has I hate when people talk shit cuzs I drink if they knew ke their children so much the same oh pajars ha I hate when people talk shit cuzs I drink if they knew their kids so equal oh piores ha

Table 4.11: Sentence 21 - Student and Teacher Normalizations - Teacher Model 1

a literal translation of ”ke”, rather leaving it out, which does not create an issue but is noteworthy. All models struggle with ”oh priores”, leading to issues with the expression ”tan igual oh piores”, causing some grammatical errors in the normalizations. Overall normalization quality is worse than the teacher model due to the issues with ”piores”. Of the transformer-tiny-sized student models of teacher model 2, model 1 does not translate ”ke” correctly, but attempts to translate the ”tan igual oh piores” expression mostly correctly, although it does not translate ”piores”. None of the other models translate ”piores” correctly, or even at all. Model 3 attempts and translates it as ”peppers”, which is a lexical error. Student model 5 also has a grammatical error in the case of ”know”, which should be ”knew”. Overall normalization quality is about the same as transformer-size models; nothing major stands out.

Teacher model 3 does not translate ”ke” correctly, as shown in 4.13; however, ”piores” is correctly translated, even though the model struggles with the ”oh” instead of ”o”,

Model	Sentence
Original	I hate when people talk shit cuzs I drink si supieran ke sus hijos tan igual oh piores ha
Native Translation	I hate when people talk shit cause I drink if they knew that their kids are the same or worse ha
Teacher 2	I hate when people talk shit cuzs I drink if they know that their children are equally worse
Students - Transformer	I hate when people talk shit cuzs I drink if they know their children so equal oh piores ha I hate when people talk shit cuzs I drink if they know that their children are so equal oh piores ha I hate when people talk shit cuzs I drink if they know that their children are as equal oh piores ha I hate when people talk shit cuzs I drink if they know your children as equal oh piores has I hate when people talk shit cuzs I drink if they know that their children are so equal oh piores ha
Students - Tiny	I hate when people talk shit cuzs I drink if they know what their children are as equal to oh piores ha I hate when people talk shit cuzs I drink if they knew their children as equally oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh peppers ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they know their children as well as oh piores ha

Table 4.12: Sentence 21 - Student and Teacher Normalizations - Teacher Model 2

which subsequently messed up the grammar with the preceding words. Normalizations of teacher model 3's transformer-sized student models are extremely similar, varying only in model 5 translating "tan igual" as "as equal" and not "so equal", which can be attributed to none of the models translating "piores", which makes it harder for the model to predict the needed case. Additionally, none of the models translate "ke" literally, but choose to leave it out, which, given the full sentence, would not be a mistake, and model 5 leaves out the "ha" at the end. Normalizations by the transformer-tiny-sized students of teacher model 3 are again extremely similar. Only model 4 has a different translation for "tan igual" going with "so the same" as opposed to the other models' "so equal". Student model 5 includes but does not translate "ke" as well as translates "sus hijos" as "her children" and not "their" children". Again, however, none of the models translate "piores" showing that the different writing for "o" is causing issues for the models.

Normalizations by the baseline 2021 transformer-sized models, shown in table 4.14

Model	Sentence
Original	I hate when people talk shit cuzs I drink si supieran ke sus hijos tan igual oh piores ha
Native Translation	I hate when people talk shit cause I drink if they knew that their kids are the same or worse ha
Teacher 3	I hate when people talk shit cuzs I drink if they knew ke their children so equal oh worse ha
Students - Transformer	I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children as equal oh piores
Students - Tiny	I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their children so the same oh piores ha I hate when people talk shit cuzs I drink if they knew ke her children so equal oh piores ha

Table 4.13: Sentence 21 - Student and Teacher Normalizations - Teacher Model 3

are very similar; models 2 and 5 include the non-translated ”ke” while the other leave it out. Model 4 translates ”hijos” as ”kids” and not ”children”, using a term which has not shown up in any student model before. This also applies to model 1’s ”alike” for ”igual”. However, no model translates ”piores” at all. Normalizations for the baseline 2021 transformer-tiny models differ more, with more issues and more generation errors, such as repetitions in models 1, 2, and 4, as well as some models leaving out the ”piores” expression completely. Model 2 turns it into the correct form ”peores”, which is interesting as models are not trained to correct spelling in Spanish.

Baseline 2023 transformer models’ normalizations, shown in table 4.15 are not correct, leaving out the ”piores” section or not translating it. However, models 1, 2, and 4 translate ”hijos” literally as ”sons”, which has not been seen in any other models. Models also have two translations for ”igual”, models 1, 3, and 4 translating it as ”equal” and model 2 translating ”same”, while model 5 cuts off its translation before that part. Translations

Model	Sentence
Original	I hate when people talk shit cuzs I drink si supieran ke sus hijos tan igual oh piores ha
Native Translation	I hate when people talk shit cause I drink if they knew that their kids are the same or worse ha
Baseline 2021 - Transformer	I hate when people talk shit cuzs I drink if they knew their children so alike oh piores ha I hate when people talk shit cuzs I drink if they knew ke their children so equal oh pyores has I hate when people talk shit cuzs I drink if they knew their children so equal oh piores ha I hate when people talk shit cuzs I drink if they knew their kids so equal oh piores has I hate when people talk shit cuzs I drink if they knew ke their kids so same oh piores
Baseline 2021 - Tiny	I hate when people talk shit cuzs I drink if they knew you your children just like, oh piores ha I hate when people talk shit cuzs I drink if they knew ke ke their children so much like, oh peores ha I hate when people talk shit cuzs I drink if they knew their children so alike I hate when people talk shit cuzs I drink if they knew if they knew if they knew their children like oh piores has I hate when people talk shit cuzs I drink if they knew their children like

Table 4.14: Sentence 21 - Baseline 2021 Normalizations

from the baseline 2023 transformer-tiny-sized models are worse than transformer-size versions, as models 1 and 4 translated ”supieran” wrongly as ”sup”. Model 1 also leaves out the ”piores” part, while models 2, 3, and 5 leave it in untranslated. Model 4 cuts off its translation after ”same”.

Tables 4.16 to 4.20 show sentence 4023 of the code-mixed test set, which was analyzed in detail, as well as the native and machine normalizations. The sentence has an almost 50/50 split between English and Spanish, separated by the Spanish ”jaja” and the Americanized Mexican Spanish term ”lonche”. Since ”lonche” is a Mexican way to spell ”lunch”, regional language differences might show in normalizations.

Teacher model 1’s normalization, shown in table 4.16, is almost completely correct, except that ”lonche” is not entirely correctly translated and was instead turned into into ”lunche”. All transformer-sized student models of teacher model 1 struggle with ”lonche” since no model has translates it. Model 1 struggles with ”haha”, translating it as ”hat”; models 1 and 5 do not catch the correct personal pronoun for ”pay”. The colloquial terms such as ”jkjk” and ”lmao” are just kept the way they are, which is correct. Normalization

Model	Sentence
Original	I hate when people talk shit cuzs I drink si supieran ke sus hijos tan igual oh piores ha
Native Translation	I hate when people talk shit cause I drink if they knew that their kids are the same or worse ha
Baseline 2023 - Transformer	I hate when people talk shit cuzs I drink if they knew what their sons were so equal oh I hate when people talk shit cuzs I drink if they knew me that their sons so same oh piores ha I hate when people talk shit cuzs I drink if they knew like their children so equal oh piores ha I hate when people talk shit cuzs I drink if they know your sons so equal oh piors ha I hate when people talk shit cuzs I drink if they knew their kids so much
Baseline 2023 - Tiny	I hate when people talk shit cuzs I drink if they sup their kids so equal I hate when people talk shit cuzs I drink if they knew their children so much like oh piores ha I hate when people talk shit cuzs I drink if they knew ke their children so same oh piores ha I hate when people talk shit cuzs I drink if they sup their children so much the same I hate when people talk shit cuzs I drink if they knew their children so equal oh piores has

Table 4.15: Sentence 21 - Baseline 2023 Normalizations

quality for the transformer-tiny-sized student models of teacher model 1 is worse than transformer-sized models: Model 1 and 4 do not translate ”jaja”, model 3 do not retain ”sis” correctly, no models translate ”lonche” and model 3 switches the last three words around as well as giving the wrong personal pronoun to the sister, namely ”he” instead of ”she” or ”you”. Attribution of the paying action is an issue in three translations; models 1, 2, and 4 attribute the paying to the writer and not their sibling. Model 5 is the closest to a correct translation, although it also keeps the ”lonche” and adds an additional ”ha” to the ”haha”.

Teacher model 2 has some issues with its normalization, shown in table 4.17. For example, ”siss” instead of ”sis”, and not translating ”jaja”. Additionally, the model is showing struggles with regional terms, since ”lonche” is not translated. In addition, ”pay” is not in the correct inflection. For this sentence, teacher model 1 produces a better normalization. Teacher model 2’s transformer-sized students do not translate ”lonche” and ”jaja” at all. Only model 5 has the correct inflection for ”pay”. The words that are translated are translated correctly, however. None of the transformer-tiny student models

Model	Sentence
Original	I love working with my sis only sometimes jaja cuando paga el lonche jkjk lmaoo
Native Translation	I love working with my sis only sometimes haha when she pays the lunch jkjk lmao
Teacher 1	I love working with my sis only sometimes haha when you pay the lunche jkjk lmaoo
Students - Transformer	I love working with my sis only sometimes hat when paying the lonche jkjk lmaoo I love working with my sis only sometimes haha when you pay the lonche jkjk lmaoo I love working with my sis only sometimes haha when you pay for the lonche jkjk lmaoo I love working with my sis only sometimes haha when you pay the lonche jkjk lmaoo I love working with my sis only sometimes haha when pay the lonche jkjk lmaoo
Students - Tiny	I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes haha when paying the lonche jkjk lmaoo I love working with my siss only sometimes haha when he pays the jkjk lmaoo lonch I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes hahaha when you pay for the lonche jkjk lmaoo

Table 4.16: Sentence 4023 - Student and Teacher Normalizations - Teacher Model 1

of teacher model 2 translate ”jaja” or ”lonche” and all have the wrong inflection and form for ”pay”. The translations are more uniform, with only model 1 being different for ”pay” instead of” paying” as well as not translating ”el lonche”.

”jaja” and ”lonche” are not translated by teacher model 3, shown in table 4.18; in this case, ”el lonche” is kept as it was before. In addition, ”paying” is the wrong form needed in this situation, as well as missing a personal pronoun. Translations for transformer-sized student models of teacher model 3 are very similar to each other, with only model 2 keeping ”el” and not translating it to ”the”. Additionally, ”jaja” and ”lonche” are not translated, and ”pay” is attributed to the wrong person for all models. The transformer-tiny-sized models’ translations of teacher model 3 are in the same situations as transformer-sized models; the difference is that model 3 is doing what model 2 did before.

The normalizations of transformer-sized baseline 2021 models, shown in table 4.19 are messy, as words are mostly translated correctly, but not in the correct form. ”pay” is

Model	Sentence
Original	I love working with my sis only sometimes jaja cuando paga el lonche jkjk lmaoo
Native Translation	I love working with my sis only sometimes haha when she pays the lunch jkjk lmao
Teacher 2	I love working with my siss only sometimes jaja when pay el lonche jkjk lmaoo
Students - Transformer	I love working with my sis only sometimes jaja when paying el lonche jkjk lmaoo I love working with my sis only sometimes jaja when pay lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkkk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when you pay the lonche jkjk lmaoo
Students - Tiny	I love working with my sis only sometimes jaja when pay el lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjkk lmaoo

Table 4.17: Sentence 4023 - Student and Teacher Normalizations - Teacher Model 2

either in the wrong form or is missing the personal pronoun. ”jaja” and ”lonche” are not translated at all, and model 4 switches the last three words of the sentence around. Overall normalization quality is worse than teacher model 1 students of the same size. The normalizations by the transformer-tiny-sized baseline 2021 models are better than the transformer-size models. Models 1 and 3 attempt to translate ”jaja” and models 1-3 have the right form and personal pronouns for ”pay”. ”lonche” is not translated at all. the baseline seems to have better translations in this case than the student models of teacher model 1.

”lonche” and ”jaja” are never translated by the transformer-sized baseline 2023 models, shown in table 4.20; model 4 translated ”sis” to ”sister”, which has not been seen before. Normalizations overall are not clean. ”pay” is always missing the correct personal pronoun, since every model except model 3 has no pronoun, and model 3 has the masculine one. Normalizations are messier, and overall, the same or worse than teacher

Model	Sentence
Original	I love working with my sis only sometimes jaja cuando paga el lonche jkjk lmaoo
Native Translation	I love working with my sis only sometimes haha when she pays the lunch jkjk lmao
Teacher 3	I love working with my sis only sometimes jaja when paying el lonche jkjk lmaoo
Students - Transformer	I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying el lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo
Students - Tiny	I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying el lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo

Table 4.18: Sentence 4023 - Student and Teacher Normalizations - Teacher Model 3

2 and 3's students of the same size. Only models 2 and 4 of the transformer-tiny-sized baseline 2023 models translate "jaja" and none translate "lonche". Model 4 has the best normalization, but otherwise, normalizations are not very uniform, and other models have issues with the attribution of "pay" as well as grammar.

Tables 4.21 to 4.25 show sentence 1216 of the independent test for detailed analysis as well as native and machine normalizations. The sentence is mostly English with some Spanish and also has some colloquial version of "that" without the "a" as well as a lowercase name. These might be issues for models. Additionally, the name "Marlene" has the suffix "-sita", which might pose additional issues to models as they have to recognize the named entity as well as the suffix and be able to separate their meanings.

Teacher model 1 does not translate and separate "marlenesita" correctly, as shown in table 4.21. "tht" and "manny" are carried over correctly, but the Spanish end of the sentence is not translated at all. The model seems to struggle with the sentence.

Model	Sentence
Original	I love working with my sis only sometimes jaja cuando paga el lonche jkjk lmaoo
Native Translation	I love working with my sis only sometimes haha when she pays the lunch jkjk lmao
Baseline 2021 - Transformer	I love working with my sis only sometimes jaja when pays the lonche jkjk Imaoo I love working with my sis only sometimes jaja when pay the lonche jkjk lmaoo I love working with my sis only sometimes jaja when pays the lonche jkjk lmaoo I love working with my sis only sometimes jaja when pays the jkjk lmaoo lonche I love working with my sis only sometimes jaja when he pays the lonche jkjk lmaoo
Baseline 2021 - Tiny	I love working with my sis only sometimes haja when you pay the lonche jjk lmaoo I love working with my sis only sometimes jaja when you pay the lonche jkjk Imaoo I love working with my sis only sometimes haja when you pay the lonche jkjk lmao I love working with my sis only sometimes jaja when paying the lonche jkklmaoo I love working with my sis only sometimes jaja when paying the lonche jkjk lmaoo

Table 4.19: Sentence 4023 - Baseline 2021 Normalization

”Marlenesita” is carried over without translation by the transformer-sized student models of teacher model 1, ”tht” and ”manny” are also carried over. The Spanish ending is partially translated by all models. Models 3 and 5 attempt to translate the word ”mato” as well, but fail to fully translate the term ”me mato” correctly. All models translate ”si no” wrongly as ”if I don’t” instead of ”if not”. Model 4 translates ”mato” as ”love”, changing the semantics of the sentence entirely. Overall, most translations are better than the teacher’s due to the attempted translations of the Spanish part. ”Marlenesita” is not translated by any student model of teacher model 1 of transformer-tiny size either, but ”tht” and ”manny” are carried over. All models again attempt to translate the Spanish part as well, but models 1 and 2 fail after ”si” and model 5 kept ”mato”. Models 3 and 4 translate it wrongly or incompletely as ”if I don’t kill”, missing the ”myself”. Normalization quality is slightly better than for the transformer-sized models, as the erroneous ”loved” translation is not present.

”Marlenesita” is not translated by teacher model 2, shown in table 4.22, and ”tht” and ”manny” are carried over. The Spanish part of the sentence is not translated at all again.

Model	Sentence
Original	I love working with my sis only sometimes jaja cuando paga el lonche jkjk lmaoo
Native Translation	I love working with my sis only sometimes haha when she pays the lunch jkjk lmao
Baseline 2023 - Transformer	I love working with my sis only sometimes jaja when pay el lonche jkjk k lmaoo I love working with my sis only sometimes jaja when pay el lonche jkjk Imaoo I love working with my sis only sometimes jaja when he pays el lonche jkjk Imaoo I love working with my sister only sometimes jaja when pay el lonche jkjk lmaoo I love working with my sis only sometimes jaja when pay el lonche jkjk lmaoo
Baseline 2023 - Tiny	I love working with my sis only sometimes aja when payel lonche jkjjk lmaoo - I love working with my sis only sometimes haha when pays the lonche jkjk lmaoo - I love working with my sis only sometimes jaja When pay el lonche jkjk lmaoo - I love working with my sis only sometimes haha when you pay the lonche jkjk lmaoo - I love working with my sis only sometimes jaja when pays the lonche jkjk lmaoo

Table 4.20: Sentence 4023 - Baseline 2023 Normalizations

For the student models of transformer size of teacher model 2, ”Marlenesita”, ”tht” and ”manny” are the same as before, ”si” is translated by all models. Models 1, 2, and 4 keep the rest of the sentence untranslated, but models 3 and 5 attempt the beginning with the correct word ”not”, leaving the rest of the sentence intact. Translations are better than the teacher model. Teacher model 2’s student models of transformer-tiny size treat ”Marlenesita”, ”tht”, and ”manny” the same as before. Models 1 and 3 have the closest correct translations so far, only wrongly attributing the ”si no” not to the rest of the sentence but to the writer. Model 3 also includes the ”lol” at the end. Models 2, 4, and 5 have varying degrees of attempted and wrong translations of the Spanish part of the sentence.

The beginning of the sentence is treated by teacher model 3 the same as before, meaning ”Marlenesita” is just carried over instead of being split up. The normalization is shwon in table 4.23. The model translates ”si” but leaves the rest of the Spanish part untouched. At this point, none of the teacher models translate or even attempt to translate this part of the sentence, which might be because of possible fine-tuning of the models not to engage with topics of self-harm. Student Models 1, 4, and 5 of transformer size of teacher model 3 have the same translation as the teacher model, model 3 kept the entire Spanish part, and model 2 is missing the ”myself” from the translation. All student

Model	Sentence
Original	Marlenesita better not open tht in front of manny or her dad si no me mato lol
Native Translation	Small/young/dear Marlene better not open that in front of Manny or her dad, if not I will kill myself lol
Teacher 1	Marlenesite better not open tht in front of manny or her dad si no me mato lol
Students - Transformer	Marlenesita better not open tht in front of manny or her dad if I don't mato lol Marlenesita better not open tht in front of manny or her dad if I don't mato lol Marlenesita better not open tht in front of manny or her dad if I don't kill lol Marlenesita better not open tht in front of manny or her dad if I don't love me lol Marlenesita better not open tht in front of manny or her dad if I don't kill lol
Students - Tiny	Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if I don't kill lol Marlenesita better not open tht in front of manny or her dad if I don't kill lol Marlenesita better not open tht in front of manny or her dad if I don't mato lol

Table 4.21: Sentence 1216 - Student and Teacher Normalizations - Teacher Model 1

models of transformer-tiny size of teacher model 3 attempt to translate the Spanish part of the source sentence, but none translate "mato", making these translations slightly worse than the transformer-sized ones.

Model 5 of the baseline 2021 transformer-sized models is the first one to not carry over "tht" correctly, as shown in table 4.24, translating it to "the door", but all models translate the Spanish part fully. Models 1, 3, and 5 are either missing the "myself" or have the wrong personal pronoun. Moreover, all models attribute the "so no" wrongly. Overall, normalizations are better than teacher model 1's transformer students, possibly showing inherited fine-tuning from teachers regarding the handling of self-harm language. If such fine-tuning did not occur during the training process it is still an example of inheritance of student models through Knowledge Distillation. Since no teacher model translates the part "si no me mato" completely, but the first set of baseline models does, this explanation seems very likely. However, this should not stop models from attempting to translate the

Model	Sentence
Original	Marlenesita better not open tht in front of manny or her dad si no me mato lol
Native Translation	Small/young/dear Marlene better not open that in front of Manny or her dad, if not I will kill myself lol
Teacher 2	Marlenesita better not open tht in front of manny or her father si no me mato lol
Students - Transformer	Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if not me mato lol Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if not me mato lol
Students - Tiny	Marlenesita better not open tht in front of manny or her dad if I don't kill me Marlenesita better not open tht in front of manny or her dad si no mato lol Marlenesita better not open tht in front of manny or her dad if I don't kill me lol Marlenesita better not open tht in front of manny or her dad if no me to lol Marlenesita better not open tht in front of manny or her dad if no me mato lol

Table 4.22: Sentence 1216 - Student and Teacher Normalizations - Teacher Model 2

expression, especially big models such as the teacher models. Of the transformer-tiny-sized baseline 2021 models, only model 4 has a very close to correct normalization, missing the "lol" and choosing the wrong word "me" instead of "myself". Models 3 and 4 attribute "si no" wrongly; models 1, 2, and 5 have varying stages of attempted translation of the Spanish part. Overall normalization quality is worse than the transformer-sized models.

Baseline 2023 transformer-sized model 5 tries for the first time to translate "Marlenesita" but translates it wrongly, as shown in table 4.25. All models of the same set also fully write out "tht" into "that" for the first time. Models 4 and 5 also capitalize "Manny" for the first time. Model 3 keeps the Spanish part the same, model 1 translated "si" but models 2, 4, and 5 have different translations of "mato" which are all wrong but new, showing multimodality, even though it is misguided in this case. However, for the transformer-tiny-sized models of baseline 2023, "Marlenesita" is again not translated at all. Model 2 leaves out "tht", model 3 translates it to "the" and models 1, 4, and 5 trans-

Model	Sentence
Original	Marlenesita better not open tht in front of manny or her dad si no me mato lol
Native Translation	Small/young/dear Marlene better not open that in front of Manny or her dad, if not I will kill myself lol
Teacher 3	Marlenesita better not open tht in front of manny or her dad if no me mato lol
Students - Transformer	Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if I don't kill lol Marlenesita better not open tht in front of manny or her dad si no me mato lol Marlenesita better not open tht in front of manny or her dad if not me mato lol Marlenesita better not open tht in front of Manny or her dad if not me mato lol
Students - Tiny	Marlenesita better not open tht in front of manny or her dad if me no mato lol Marlenesita better not open tht in front of manny or her dad if I don't mato lol Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if not me mato lol Marlenesita better not open tht in front of manny or her dad if not me mato lol

Table 4.23: Sentence 1216 - Student and Teacher Normalizations - Teacher Model 3

late it as "this", which is new. All models except model 3 leave the Spanish part after "si" untouched, only translating "si" to "if". However, model 2 turns "mato" into "mate" and model 5 incorporates "lol" with one l into "mato". Overall normalization quality is not a lot worse or better than transformer-sized models.

Tables 4.26 to 4.30 show sentence 5047 of the independent test for detailed analysis as well as native and machine normalizations. The source sentence is almost completely English but with one Spanish word, mirroring the first sentence of the analysis: Sentence 8. Additionally, it has one colloquial written expression in "uggghhhh" as well as an exclamation mark.

Teacher model 1 produces a perfect normalization for the source sentence, shown in table 4.26. Normalizations by transformer-sized student models of teacher model 1 are mostly correct. Model 3 produces a correct normalization for all non-colloquial words, while all other models seem to be confused by the colloquial expression and change "I" to

Model	Sentence
Original	Marlenesita better not open tht in front of manny or her dad si no me mato lol
Native Translation	Small/young/dear Marlene better not open that in front of Manny or her dad, if not I will kill myself lol
Baseline 2021 - Transformer	Marlenesita better not open tht in front of manny or her dad if I don't kill you lol Marlenesita better not open tht in front of manny or her dad if I don't kill myself lol Marlenesita better not open tht in front of manny or her dad if I didn't kill lol Marlenesita better not open tht in front of manny or her dad if I don't kill myself lol Marlenesita better not open the door in front of manny or her dad if I don't kill him
Baseline 2021 - Tiny	Marlenesita better not open tht in front of manny or her dad if not me mato lol Marlenesita better not open tht in front of manny or her dad if no me mato lol Marlenesita better not open tht in front of manny or her dad if I don't mato lol Marlenesita better not open tht in front of manny or her dad if I don't kill me Marlenesita better not open tht in front of manny or her dad if no me mato lol

Table 4.24: Sentence 1216 - Baseline 2021 Normalizations

"and". The source sentence's English parts are preserved very well otherwise, but models vary the number of letters in the "uggghhhh" expression, with only models 2 and 4 having the correct ones. Normalizations by the student models of teacher model 1 of transformer-tiny size are almost correct, "I" has again been changed to "and" by all models now. The same issues with the colloquial expression arise again, since only model 2 has the correct version.

Teacher model 2 produces an almost perfect normalization, shown in tabel 4.27, only generating one "h" too many in "uggghhhh". All transformer-sized models of teacher model 2 also produce almost perfect normalizations, but only model 1 has the correct version of "uggghhhh", which proves to be an issue for some models, teacher as well as students. Out of the transformer-tiny-sized student models of teacher model 2, only models 2 and 3 have correct translations for "prostituta", and out of those two, only model 2 has the correct version of "uggghhhh" and a full sentence since model 3 has a nonsense translation where it keeps repeating the "h" in "uggghhhh". All models keep the correct personal pronoun, however, for "I". Overall, transformer-tiny models are worse

Model	Sentence
Original	Marlenesita better not open tht in front of manny or her dad si no me mato lol
Native Translation	Small/young/dear Marlene better not open that in front of Manny or her dad, if not I will kill myself lol
Baseline 2023 - Transformer	Marlenesita better not open that in front of manny or her dad if no me mato lol Marlenesita better not open that in front of manny or her dad if I can't get out of here. Marlenesita better not open that in front of manny or her dad si no me mato lol Marlenesita better not open that in front of Manny or her dad if I don't mind. Marlene's still better not open that in front of Manny or her dad if not me must lose.
Baseline 2023 - Tiny	Marlenesita better not open this in front of manny or her dad if no me mato lol Marlenesita better not open in front of manny or her dad if no me mate lol Marlenesita better not open the front of manny or her dad if me kill him Marlenesita better not open this in front of manny or her dad if no me mato lol Marlenesita better not open this in front of manny or her dad if no me matol

Table 4.25: Sentence 1216 - Baseline 2023 Normalizations

than transformer-sized models.

Apparently, teacher model 3 treats "uggghhhh" as noise and replaces it with a grammatically correct ",", shown in table 4.28, which does not change the semantics of the sentence, but the pragmatics of it. Otherwise, the normalization is, expectedly, perfect. Only student models 1, 4, and 5 of transformer size of teacher model 3 translate "prostituta" and model 5 also exchanges "uggghhhh" with "," while all other models have varying versions of "uggghhhh" with the wrong number of letters. All models keep "I", but the normalization quality is worse than any other transformer-sized models so far. Normalizations by the transformer-tiny-sized student models of teacher model 3 are better than the transformer-sized models since all models translate "prostituta" and model 3 also has the right number of letters for "uggghhhh". Only model 5 deletes this expression, probably seeing it as noise.

Model 1 of the transformer-sized baseline 2021 models does not translate "prostituta" and model 2 changes "uggghhhh" to "," again, shown in table 4.29. Model 5 has a perfect normalization. Overall, normalization quality is slightly higher than the respective teacher

Model	Sentence
Original	When he called her a prostituta uggghhhh i just wanted to punch the fuck out of him !
Native Translation	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him !
Teacher 1	When he called her a prostitute uggghhhh and just wanted to punch the fuck out of him !
Students - Transformer	When he called her a prostitute uggghhhhh and just wanted to punch the fuck out of him ! When he called her a prostitute uggghhhh and just wanted to punch the fuck out of him ! When he called her a prostitute uggghhh i just wanted to punch the fuck out of him ! When he called her a prostitute uggghhhh and just wanted to punch the fuck out of him ! When he called her a prostitute uggghhhhhhhh and just wanted to punch the fuck out of him !
Students - Tiny	When he called her a prostitute uggghhhh and just wanted to punch the fuck out of him! When he called her a prostitute uggghhhhh and just wanted to punch the fuck out of him ! When he called her a prostitute uggghhhhhh and just wanted to punch the fuck out of him ! When he called her a prostitute uggghhhh and just wanted to punch the fuck out of him ! When he called her a prostitute uggghhhh and just wanted to punch the fuck out of him !

Table 4.26: Sentence 5047 - Student and Teacher Normalizations - Teacher Model 1

model 1 student models due to keeping ”I” in all versions. Transformer-tiny-sized baseline 2021 models 2 and 4 do not translate ”prostituta” correctly, showing even one new, albeit still wrong, version in ”prostitative” in model 4. All models except model 3 have the correct version of ”uggghhhh” and kept the ”I” as well.

For Baseline 2023 models of transformer size, shown in table 4.30, model 5 does not translate ”prostituta” and models 1 and 3 delete or exchange ”uggghhhh” for ”,”. Overall, the quality of normalizations is not very different from student models from teachers 2 and 3. Baseline 2023 transformer-tiny-sized model 4 do not translate ”prostituta” and only models 3 and 4 keep the ”uggghhhh” while the others exchange it for different terms and words. Model 3 also keeps the correct number of letters, making its normalization a perfect one. Generally, these normalizations are messier than the corresponding student models’.

The analysis shows that for sentence 8, all student models, as well as teacher models

Model	Sentence
Original	When he called her a prostituta uggghhhh i just wanted to punch the fuck out of him !
Native Translation	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him !
Teacher 2	When he called her a prostitute uggghhhhh i just wanted to punch the fuck out of him!
Students - Transformer	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhhhhhh I just wanted to punch the fuck out of him!
Students - Tiny	When he called her a prostituta uggghhhhhhhhhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhhhhhhhhhhhhhhhhhhhh-hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh When he called her a prostituta uggghhhhhh I just wanted to punch the fuck out of him! When he called her a prostituta uggghhhhhhh I just wanted to punch the fuck out of him!

Table 4.27: Sentence 5047 - Student and Teacher Normalizations - Teacher Model 2

and baseline models, have difficulty with colloquial expressions without a literal translation such as ”mierda con cojones”. However, almost all models are able to preserve the pragmatics of the source sentence, showing ability in Natural Language Understanding and being able to adapt the normalization accordingly. Common errors are wrong word choices for ”con cojones” instead of translating it or carrying them over, something that was more common in baseline models compared to student models.

For sentence 1110, tiny models seem to struggle more with keeping the named entity intact. In one case, the student models even produce a correct normalization even though their teacher does not. Moreover, often normalizations attribute the touching to the wrong person. This might be because of missing context clues or because of the different language versions that are present in the training and code-mixed test sets.

Source sentence 21 is quite difficult with multiple typos and written colloquial versions of words, ”cusz” instead of ”because”, ”oh” instead of ”o”, and ”piores” instead of ”peores”.

Model	Sentence
Original	When he called her a prostituta uggghhhh i just wanted to punch the fuck out of him !
Native Translation	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him !
Teacher 3	When he called her a prostitute, I just wanted to punch the fuck out of him!
Students - Transformer	When he called her a prostitute Uggghhh I just wanted to punch the fuck out of him! When he called her a prostituta uggghhhhh I just wanted to punch the fuck out of him! When he called her a prostituta uggghh I just wanted to punch the fuck out of him! When he called her a prostitute Uggghh I just wanted to punch the fuck out of him! When he called her a prostitute, I just wanted to punch the fuck out of him!
Students - Tiny	When he called her a prostitute uggghh, I just wanted to punch the fuck out of him! When he called her a prostitute ugggghhhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him! When he called her a prostitute ugghhhh I just wanted to punch the fuck out of him! When he called her a prostitute I just wanted to punch the fuck out of him!

Table 4.28: Sentence 5047 - Student and Teacher Normalizations - Teacher Model 3

All of this creates issues for all transformer and transformer-tiny models, as none of them produced a correct normalization, while teachers 2 and 3 are able to get close. This is likely due to their overlapping information from the number of supported languages. Interestingly, not even Knowledge Distillation could amend the issues. However, student models are more likely to just carry over the words they do not translate, while baseline models leave them out more often.

Sentence 4023 shows the differences between Mexican Spanish and Catalan Spanish clearly and how models can struggle with it, as no models translates ”lonche” correctly. Only teacher model 1 is able to do so approximately, translating it to ”lunche”. All other models fail to translate it at all. Additionally, quite a few models struggle with translating ”jaja” as well as choosing the correct form for ”pay”, which might be connected to them not recognizing ”lonche”. Interestingly, for baseline models 2021 version, the normalization quality is better for the tiny models, while for the corresponding student models, it is the

Model	Sentence
Original	When he called her a prostituta uggghhhh i just wanted to punch the fuck out of him !
Native Translation	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him !
Baseline 2021 - Transformer	When he called her a prostituta uggghhh i just wanted to punch the fuck out of him! When he called her a prostitute, I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him!
Baseline 2021 - Tiny	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him! When he called her a prostituta uggghhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhhhh I just wanted to punch the fuck out of him! When he called her a prostitative uggghhhh I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him!

Table 4.29: Sentence 5047 - Baseline 2021 Normalizations

other way around. Overall, however, student models produce better normalizations even though they were not the correct ones.

Source sentence 1216 has references to self-harm, which might affect normalization quality as none of the teacher or student models translate this part of the sentence, but the baseline models often do. Also, none of the models, baseline teacher or student, translate "Marlenesita" correctly, instead treating it like a full named entity. Moreover, none of the student or teacher models correct the error in "tht" to "that"; only baseline 2023 models do. For the student models, this might be an example of inherited literal translation or preserving words. This sentence shows very well how student models can learn intrinsic training specifications of teacher models or can be influenced by their teacher's embedded information.

Source sentence 5047 does not pose a problem for any teacher model. One student for teacher 2 has a generation error, but otherwise, student models as well as teacher models normalize the sentence mostly correctly. Models show difficulty in retaining the colloquial term "uggghhhh" and sometimes leave it out. Also, tiny baseline models have two wrong

Model	Sentence
Original	When he called her a prostituta uggghhhh i just wanted to punch the fuck out of him !
Native Translation	When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him !
Baseline 2023 - Transformer	When he called her a prostitute, I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhh I just wanted to punch the fuck out of him! When he called her a prostitute I just wanted to punch the fuck out of him! When he called her a prostitute Uggghhhhhhhh I just wanted to punch the fuck out of him! When he called her a prostituta uggghhh I just wanted to punch the fuck out of him!
Baseline 2023 - Tiny	When he called her a prostitute to talk about it just wanted to punch the fuck out of him! When he called her a prostitute up and I just wanted to punch the fuck out of him! When he called her a prostitute uggghhhh i just wanted to punch the fuck out of him! When he called her a prostituta ugghhhh I just wanted to punch the fuck out of him! When he called her a prostitute fighthhh I just wanted to punch the fuck out of him!

Table 4.30: Sentence 5047 - Baseline 2023 Normalizations

normalizations, while these kinds of errors are absent for all student models.

Chapter 5

Discussion

As seen in the results section, for transformer-sized models, only models 2 and 3 students show statistical significance towards baseline models for the Language Normalization task. However, model 1 students show a more consistent performance compared to baseline with lower standard deviation. Since all students improve their performance when dealing with code-mixed data compared to their teachers, the results demonstrate the effectiveness of Knowledge Distillation for increasing noise robustness compared to teacher models. The fact that model 1 transformer students do not show statistical significance might be due to the training set being different than the training set for teacher model 2 and 3 students, and shows a strongly skewed distribution of es-en to en-en sentence pairs. This most likely has an effect on student model convergence, as overfitting might set in earlier than for other student models with a more balanced training set. Unfortunately, the log does not give detailed information on which part of the validation set started performing worse. This can be amended by validating the involved languages separately instead of a single file. It would also be interesting to see if anything changes when using the same dataset for all students instead of following the teacher models' versions of training datasets. However, in this thesis, the focus is on the increase in student performance when trained closely to how their teacher models were trained as the focus is on teacher-student-baseline triplets.

Another likely reason for students of teacher model 1 performing differently from students of teachers 2 and 3 is that the teacher model is not trained to work with English as an input language, a task that the student models are trained for. Interestingly, however, despite this, when looking at transformer-tiny-sized models, all student models show statistical significance when compared to baseline models, even model 1 students. This is especially interesting given that a smaller model size might be more likely to overfit with a skewed dataset such as the one used for training teacher model 1 students. This would mean that the example of teacher model 1 students of transformer size could be

an instance of underfitting for English-English while Spanish-English was the strongest indicator for stopping the training process.

Even though model 1 tiny students do not significantly outperform their teacher model, they do significantly outperform the baseline model trained on the same dataset, showing the effectiveness of Knowledge Distillation even in significantly smaller models. It might be that internalized similarities between Occitan, Spanish, and Portuguese were so compressed and carried over to the student models that they caused the increase in robustness. Either this, or the theory of the model originally underfitting for English-English is true and the student models of tiny size are able to make more use of the dataset for training. The performance increases of Knowledge Distillation are smaller in tiny models, which shows in the comparison between transformer vs tiny models, which makes sense considering the compression of information that takes place with the reduced capacity of smaller models. Nevertheless, the fact that teacher models 2 and 3 support as many languages as they do, and still the tiny models show significant increases in their noise handling, is interesting and suggests a very effective compression of embedded information, both compared to teacher models as well as baseline models.

However, as the normalization direction is SpaEng -> Eng, it might also be that the teacher model 1 simply interpreted the English parts as noise to be carried over and not to be translated, making specific training for this direction for the teacher model unnecessary. In theory, this means that it could be possible to train a teacher model for source languages only that need to be normalized into English and have the students learn to keep English parts of a sentence as is, as is the case in a lot of teacher translations of teacher 1 in the analyzed sentences. The student models' capabilities in noise handling might therefore be because the teacher model is good at identifying the parts of a sentence that are to be simply carried over to the normalized version. This theory is supported by the fact that the other teacher models and their respective student models are equally capable of retaining English parts of a sentence while training time for teacher model 1 is significantly lower than for teacher model 2. However, teacher model 3 trained for even shorter time which raises questions regarding the possible effectiveness of using a bigger number of target languages than source languages. This, however, would have to be investigated separately.

When it comes to student models' performance regarding the separate tasks they were trained for, namely translation of Catalan Spanish-to-English and English-to-English cleaning, student models tend to mimic teacher model performance for both tasks and rarely improve on it. The exception to this are the student models of teacher model 1, which makes sense given that teacher model 1 is not trained for this task. Nevertheless, it is very interesting that teacher model 1 performs better for English-English than for

Spanish-English. This sometimes causes the models to be worse than the baseline models, but could be amended by introducing natural translations into the training data after it has been translated by the teacher model and possibly achieving higher results with the augmented dataset. However, this might also affect the model's ability to handle noise and code-mixing, so further testing would be necessary. Still, the results show that student models to retain comparable teacher performance even in tiny models, showing strong capabilities for effectively compressing embedded information, not just for noise robustness. Interestingly, English-to-English scores are better for teacher 1 student models than their teacher model. This is surprising as the teacher model is not trained for this kind of task, but could be explained with the aforementioned approach of interpreting English sentence parts are parts to be retained for the normalized sentence. What is even more interesting is that the teacher model itself performed better in en-en than es-en. This might be, however, because the en-en dataset requires the model to mostly keep the sentences the same but with slight variations, which further supports the theory that the teacher model is good at identifying which parts of a sentence to ignore. This skill then shows up again in the analyzed sentences which have a higher amount of English, as the teacher model is able to simply carry the English parts over to the normalized sentence.

As expected, training times and translation times are lower for student models than for teacher models. Teacher training times are very high and vary a lot as well, most likely due to the different number of languages, as teacher model 2 shows the longest training time. Interestingly, student models for teacher model 2 do not have the longest average training time out of all student models, suggesting an effective compression of embedded knowledge. A possible reason might be overlapping information encoding of closely related languages, which might be less present in teacher models 1 and 3 due to the lower number of input languages which also means less "noise" of possibly unimportant and unrelated languages. Of course, teacher training times cannot be compared to their student model training times, as the teachers were trained with very different datasets. However, it is possible to gain some insight in the effectiveness of Knowledge Distillation and implications to training times of students which seem to be entirely independent from their teachers, which is signified by teacher model 2's students not training the longest. This would mean that while it takes longer to train a teacher model with more languages, it might be beneficial regarding training and translation times, as the condensed information seems to prove very useful for the resulting student models, and it does not cause longer training times for student models.

Furthermore, the baseline models for the 2023 version of the dataset show the highest training time for both model sizes. Since the language ratio in the utilized dataset is more balanced than for the baseline 2021 version, this might explain at least the baseline models'

training duration differences, as models would converge more quickly to avoid overfitting on the overrepresented language. This, however, does not explain why transformer-tiny baseline 2021 models take longer than their transformer-sized versions to converge, as the more limited model capacity would mean that the overrepresented language is more likely to overfit quickly. Unless, of course, the limited capacity allows the underrepresented language to have a larger effect during training which prolongs convergence. Given that the transformer-tiny models perform statistically significantly worse for Spanish-English and Spanglish-English and significantly better for English-English, this does seem to be the case, however the more pronounced effect of the English-English part of the dataset means that not only does convergence take longer, the overall model performance suffers for noise handling, highlighting the need for well designed datasets for training models.

The process of Knowledge Distillation also seems to have a positive effect on training times when compared to baseline models, as all student models of all sizes improve their training time by at least 25.1% for transformer-sized models and 17.3% for transformer-tiny-sized students. This, of course, also means that the amount of needed resources for training these models is considerably lower, making Knowledge Distillation a reasonable choice for a more sustainable approach to using neural networks. The fact that transformer-sized student models of teacher model 1 show the lowest gains for training time makes sense given the issues with the training set. Interestingly, the situation is the opposite for transformer-tiny models, where teacher model 1's students have the highest decrease in training time. Given that the students outperform their baseline in the normalization as well as the Spanish-English translation task and the baseline models only barely significantly outperform the student models in the English-English task, it seems that a smaller model size for this specific training data is a better fit and has to most to gain from Knowledge Distillation. It is possible that students of teacher model 1 showed a bigger decrease in training time if they were trained with the same data as the other student models. As it is, there seems to be a relation between the number of supported languages for teacher models and the gains of student model training times, as students of teacher model 3 show the best increase for transformer-size, which is not true for the smaller models. The stronger decrease of training time for teacher 3's students compared to teacher 2's students might be linked to the number of supported languages, as mentioned in chapter 4, which makes it harder for very small models to encode information effectively, making a smaller number of languages, such as teacher model 3 has it, a more convenient setup. What would be interesting to see, however, would be whether the training time changes if an approach were to be used that used several teacher models with different languages which do not overlap. Because the teacher models do not have much embedded information by themselves, this might make it easier for student models

to adapt their own parameters accordingly. At the same time it is equally possible that training time is increased as the models might introduce conflicting information which extends the time to converging.

In addition to faster training times, student models also show a much higher inference speed, transformer-sized models reducing the teacher inference speed by 98% on average and transformer-tiny students showing a reduction of 99.1% compared to teacher models. In comparison, baseline transformer models reduce the inference time by 97.65% and transformer-tiny models by 98.55%. Though the difference between baseline and student models is not big, this would be important to take into account as their speed increase, if not substantial, but coupled with their increased noise robustness, means they are able to handle more input better than a baseline model and therefore would be more viable for more widespread use.

With the student results and their respective teacher performances, it is important to also note that neither of the teacher models finished training because they converged; rather, the training was stopped manually at a certain point. Taking this into account, using Knowledge Distillation for a fully trained and converged teacher model, training a smaller model could possibly reduce training duration even more and improve performance further. This would have to be tested with converged teacher models in order to be confirmed, however. Nevertheless, it is possible that the results might look very different if, for example, teacher model 1 converged at a better performance overall than teacher model 2 or 3, or if teacher model 3 converged at a better performance than the other two.

As seen from the results, further compressing models into the transformer-tiny version improves training and inference time even further. Resource-wise, therefore, using KD to train a model a small as possible seems to be the best option since performance is still comparable to teacher models, with occasional significently improved or worsened performance on single tasks, but consistently significant performance increases for Language Normalization tasks. However, as the sentence analysis shows, it might introduce new issues with interpreting unusual sentences or expressions. After all, transformer-sized students outperform their transformer-tiny-sized counterparts significantly in almost every task and scenario presented in this thesis. It is also important to note, that while the difference between student model and baseline model training time is substantial, and even present for inference times, this difference does still diminish with further compression which also shows in performance as in the case of model 1 and 2 performance is significantly worse in tiny models than in transformer models and in the case of model 3, while not significantly worse, still worse. Mevertheless, gains over baseline models are present and quite substantial as well. The actual use of compressing models even smaller might only be present when the teacher model has an appropriate amount of condensed

information, as tiny students for model 2 show worse performance than tiny students for model 3, even though it is not significant.

Detailed differences in model performance are sadly not easily available, as the gold sentences are not available for the used independent dataset and with the website inaccessible there is no way of obtaining them at this point. Only BLEU scores are provided, which in themselves have to be seen with care due to their nature and also because the student and baseline models are usually trained on a lot less languages than the teacher models, which likely has an effect on their performance as well as shown by Aharoni et al. (2019) where a degradation in performance was recorded for increasing numbers of languages in Machine Translation models. As described in section 3.5, a detailed sentence requires to the sentences to be chosen with care, as the code-mixed test set is incredibly noisy, having a lot of colloquial language, hashtags, links, and emojis. This level of additional noise, though naturally occurring and therefore representative of real-life situations, certainly also has an effect on normalization quality and makes it difficult to attribute normalization fails clearly to the code-mixing or general noise. Since no additional information about the code-mixed test set is known, except for the source sentences and BLEU scores, it is very difficult to check even the predicted sentences for noremalization quality. This effect of additional noise can be further investigated by training a model that is specifically trained for noise handling and excluding code-mixing from the training, and seeing how the model's performance changes.

That said, the checked sentences for detailed analysis all show different possible issues that translation models might have to deal with should they be used for this task regularly. The detailed analysis shows, especially in the case of sentence 1216, that possible issues of inheriting embedded information and biases can stop student models from translating some topics, such as self-harm, which might be intended for some uses but is something to be kept in mind if this approach is to be used more widely. Generally, tiny models do make a lot more translation mistakes for the analyzed sentences compared to transformer models, but students often also make different errors compared to teacher models and other student model sizes, which shows that, despite Knowledge Distillation, smaller models still have reduced capabilities. This is most present in sentence 21, as teacher models 1 as well as all its student and baseline models are unable to correct the typo in "piores" and interpret the word correctly, while teacher models 2 and 3 did. This also shows that including more languages in a model's range increases its capabilities to correctly handle typos, which is something that seems to be missing from students and baseline models alike, at least in this example, and would be worth investigating in further research where students might be trained for all the languages their teacher models support. Nevertheless, the fact that student models tend to simply keep unknown

phrases instead of trying to translate them, and possibly create wrong translations, as seen in numerous wrong translations by baseline models, remains. The detailed analysis also shows very well how important it is to choose training data carefully. Not only does it show in how differently students of teacher model 1 perform compared to teacher model 2 and 3, and even the baseline models showing differences in performance, Sentence 4023 shows this very well with its use of ”lonche”, which is a Mexican Spanish way that is Americanized for saying ”lunch”. This term does not exist in European Catalan Spanish, which is what the training and Tatoeba test set is made up of. Moreover, student and baseline models struggle more with colloquial expressions than teacher models, showing again the limitations of smaller models that could not be circumvented even by Knowledge Distillation in this example.

Given the fact that the performance of transformer-tiny models is overall lower than their transformer-sized counterparts, whether or not further compression of larger teacher models is useful for large-scale use cases seems to depend on the importance of accuracy vs inference and training time. A bigger model might be worth the additional resources needed for training an dinference in one case, while speed might be of the essence in another. One issue that causes the performance difference might also be the heavy compression of the teachers' embedded information, which might be too much to handle for the transformer-tiny models. This shows repeatedly in differences between transformer and transformer-tiny-sized students and how training and inference times relate to different model sizes and their baseline counterparts. The situation might be different when using a sequential Knowledge Distillation approach where the teacher model trains smaller student models, for example, transformer-sized, and these student models in turn are used to train even smaller models, for example, transformer-tiny-sized. In this case, an ensemble training approach could be used as well, which possibly allows for compression of information without significant loss due to reduced model capabilities. For example, one student model could be trained for a subset of languages the teacher model supports, while another is trained on a different subset so that all together they can come together to ensemble train the next iteration of student models that support all languages the original teacher model supports.

Taking the difference of performances across all student models into account and relating them to their teacher models' language capabilities reveals that while the teacher 1 students are not outperforming other teachers' students in the English-to-English task, they do consistently outperform the other students when it comes to es-en translation, both in transformer and transformer-tiny models. This even applies to baseline models for the 2021 data set, while the reverse is true for teacher models 2 and 3, students, and their respective 2023 baseline models. The difference in the English-to-English task is

most likely due to the fact that teacher model 1 is not trained to take English as an input language, while the other two teachers are. However, the performance difference in Spanish-to-English translation shows very well how the number of involved languages for a teacher model can influence the student performance for the translation direction, as shown in Aharoni et al. (2019). Interestingly, the higher number of supported languages seems to cause the student models of teacher models 2 and 3 to show better noise and code-mixing handling capabilities. This, most likely, stems from the amount of information that was compressed from the teacher models, as the student models can make use of overlapping information due to language similarities present in the teacher models while also being able to deal with a wider variety of input token combinations. Taking all this into account, it seems that not only compression level of the models play a big role in the student model's capabilities, but also the number of languages and encoded information of the teacher model. Knowing this, it should be possible to design a teacher model for specific purposes and situations and then use this to train student models that are used in the a way of specialized tasks, for example, Language Normalization, as Knowledge Distillation has shown to be quite effective. Additionally, it is important to make sure the training data is suited for the task as well, which is shown in teacher model 1's students' performances, as one reason for the lower code-mixed performance might be the skewed nature of the training set. If it is possible to have training sets for specific tasks and situations as well, it is possible to effectively train and use a wide range of student models that share their teacher's capabilities but improve on certain needed aspects. In the case of Language Normalization, a balanced training set as well as a teacher model that is capable to not only work with the languages the student model is needed for, but also related and even unrelated languages in order to improve noise handling even more would be a very useful tool in order to achieve optimal model performance while making sure that resources are used responsibly. Given all this information, the first research question asking whether student models can show improved performance in Language Normalization tasks after Knowledge Distillation compared to their teacher models can be answered with yes. However, the second research question whether stuydent models also outperform baseline models which are trained from scratch when trained on the same training data is not as clear, since model performance generally for the code-mixed Language Normalization task seems to generally improve, but performances on separate tasks such as English-English cleaning and paraphrasing and Spanish-English translation depend on the teacher model performance and can therefore not be answered with a general "yes".

However, the fact that the difference between transformer and transformer-tiny models for teacher model 3 is not significant, and the fact that the teacher models did not converge

before their training finished, seems to suggest that with proper convergence and dataset composition, this might change. In addition, it might be useful to explore serial Knowledge Distillation to counteract diminishing model capabilities at smaller model sizes.

Several limitations of my research have to be taken into account, however.

First of all, the code-mixed test set that is used for the Language Normalization task is very noisy, so further research has to be done in order to confirm that the effect of Knowledge Distillation is noticeable for code-mixing and not general noise handling. This can be further investigated by training models that are specifically designed to deal with noise instead of clean language and comparing performance to the student models as well as training models on a purely code-mixed dataset without any other noise.

Secondly, in order to allow for a more detailed comparison between different student model groups training with only one dataset would be beneficial. For the purpose of this thesis, the focus was more on making comparisons between student and baseline models that were trained on similar datasets to their teacher in order to gain a more immediate comparison how performance on the Language Normalization task changes after Knowledge Distillation.

Thirdly, the student models are not trained on all the languages their teachers were trained one. This could very well skew performances, as Aharoni et al. (2019) has shown how the number of involved languages can affect a model's performance. Either students could be better than observed due to the shared embedded linguistic properties of multiple languages, or performance could be worse due to the reduced capacity of smaller models. This issue can be addressed by training new student models that support all their teachers' languages. Due to technical and time limitations, however, this is not possible for this thesis.

Furthermore, performance might be different if the training set language fit the code-mixed test set languages more exactly, as the training set is based on Castilian Spanish, while the test set is based on mostly Mexican Spanish. Eliminating differences between these languages might improve the accuracy of the results further. Even so, the comparison between student and teacher models has to be seen with caution, as the datasets used for training are different sinc ethe student model datasets do not include backtranslated data or bible translations.

Moreover, the experiment which was performed to assess whether creating my own gold sentences with a Machine translation model might have yielded different results if: The model used for translations was not an Opus model which are generally trained on similar data, and if the model had been trained for handling code-mixed data as it is likely that the teacher models make similar mistakes to the model used for creating the translations, which would obviously have an effect on the final BLEU scores.

Finally, it is also possible that performance of student models could be improved even further if natural training data were mixed with Knowledge Distillation training data as the augmented dataset might exhibit features critical for not only improving noise handling, but also retaining the performance of teacher models for individual tasks.

Chapter 6

Conclusion

In this thesis, I investigate the effects that sentence-level Knowledge Distillation has on Machine Translation models for Language Normalization tasks when dealing with Code-mixed input. To do this, three teacher models are selected, which all support different numbers of source and target languages. For each of these models, which are all of transformer-big architecture, five student models are trained of transformer architecture and five models of transformer-tiny architecture. In addition, since the teacher models uses two different training sets, for each training set five baseline models of transformer architecture as well as 5 models of transformer-tiny architecture are trained. All models are scored on their abilities to translate from Spanish-to-English, their ability to clean up English input, and finally on the code-mixed test set. Performance is recorded in BLEU scores, and statistical analysis is performed comparing the teacher models to their students as well as corresponding baseline models, the student models to their respective baseline models of the same size, and also the student models and baseline models to each other's smaller versions. Furthermore, six sentences are chosen from the code-mixed test set, which all present unique challenges of the involved languages as well as special features, such as colloquial expressions, spelling errors, and dialectal expressions.

The analysis of the results reveals that every student model of transformer size outperforms their teacher model significantly in the Language Normalization task, and almost every student model of transformer size outperforms their respective baseline model significantly. Additionally, almost every transformer-tiny student outperforms their teacher significantly, and every transformer-tiny student outperforms their baseline models significantly. At the same time, individual task performances for Spanish-to-English translation and English cleanup are usually comparable to the students' teacher models, which influences their performance compared to their baseline models. Finally, transformer-sized students perform significantly better than their tiny counterparts in two of the three student model sets, with the third set barely passing significance in its difference. While still

better than their baseline counterparts of the same size, further compression of models does not improve performance further.

Sentence analysis shows that student models' reduced model capacity generally keeps them from correcting their teacher's errors in normalization. Furthermore, student models struggle with typos as well as out-of-vocabulary words such as the Mexican Spanish-specific "lonche". Analysis also reveals that the inheritance of their teacher's embedded information may keep student models from reaching their full potential in certain aspects, as no teacher models translates mentions of self-harm, which is transferred to student models, while baseline models attempt and in some cases succeed in translation.

For the purpose of this thesis, based on the presented results, it can be concluded that Knowledge Distillation does have a beneficial effect on model robustness when dealing with code-mixing even if the data used for training does not contain code-mixed language. Furthermore, it seems to have a positive effect on noise handling in general, since the code-mixed test set is very noisy and student models still outperformed their teachers and their baseline models. The results do not, however, imply that further compression of models improves performance even further and performance on separate tasks even if the models are trained for them depends on the teacher model performance.

In sum, while there is further research to be done to confirm these results conclusively, Knowledge Distillation is a useful tool for improving model handling of code-mixing in Language Normalization tasks while keeping comparable performances for individual tasks. The improvement in resource needs and performance makes Knowledge Distillation an ideal candidate to tackle the issue of resource-intensive large translation models on a large scale. However, more extreme compression of models might be dependent on the requirements of the task, whether a smaller and faster model is needed or a more accurate performance.

Bibliography

Gustavo Aguilar, Sudipta Kar, and Thamar Solorio. Lince: A centralized benchmark for linguistic code-switching evaluation. *arXiv preprint arXiv:2005.04322*, 2020.

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

Mikko Aulamo, Sami Virpioja, and Jörg Tiedemann. OpusFilter: A configurable parallel corpus filtering toolbox. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 150–156. Association for Computational Linguistics, July 2020. doi: 10.18653/v1/2020.acl-demos.20. URL https://www.aclweb.org/anthology/2020.acl-demos.20.

Arindam Chatterjee, Chhavi Sharma, Yashwanth Vp, Niraj Kumar, Ayush Raj, and Asif Ekbal. Lost in translation no more: Fine-tuned transformer-based models for codemix to english machine translation. In *Proceedings of the 20th International Conference on Natural Language Processing (ICON)*, pages 326–335, 2023.

Mrinal Dhar, Vaibhav Kumar, and Manish Shrivastava. Enabling code-mixed translation: Parallel corpus creation and mt augmentation approach. In *Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing*, pages 131–140, 2018.

Philip Gage. A new algorithm for data compression. *C Users J.*, 12(2):23–38, February 1994. ISSN 0898-9788.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. Knowledge distillation from a stronger teacher. *Advances in Neural Information Processing Systems*, 35: 33716–33727, 2022.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

Wandri Jooste, Rejwanul Haque, and Andy Way. Knowledge distillation: A method for making neural machine translation more efficient. *Information*, 13(2):88, 2022.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. Marian: Fast neural machine translation in C++. In Fei Liu and Thamar Solorio, editors, *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-4020. URL https://aclanthology.org/P18-4020/.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition, 2025. URL https://web.stanford.edu/~jurafsky/slp3/. Online manuscript released January 12, 2025.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1317–1327, 2016.

Ágnes Melinda Kovács. Early bilingualism enhances mechanisms of false-belief reasoning. *Developmental science*, 12(1):48–54, 2009.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*, 2018.

Ayodele Marvellous, Sudarshana Karkala, Sazzad Hossain, Mahendra Krishnapatnam, Ankur Aggarwal, Zarif Zahir, Harshad Vijay Pandhare, and Varun Shah. Low-resource code-mixing translation using multilingual transformers and adapter layers. 2025.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5191–5198, 2020.

Giovanni Molina, Fahad AlGhamdi, Mahmoud Ghoneim, Abdelati Hawwari, Nicolas Rey-Villamizar, Mona Diab, and Thamar Solorio. Overview for the second shared task on language identification in code-switched data. *arXiv preprint arXiv:1909.13016*, 2019.

Li Nguyen, Christopher Bryant, Oliver Mayeux, and Zheng Yuan. How effective is machine translation on low-resource code-switching? a case study comparing human and automatic metrics. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 14186–14195, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.893. URL https://aclanthology.org/2023.findings-acl.893/.

Leiyu Pan, Deyi Xiong, et al. An empirical study on the robustness of massively multilingual neural machine translation. *arXiv preprint arXiv:2405.07673*, 2024.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. The university of edinburgh's neural mt systems for wmt17. *arXiv preprint arXiv:1708.00726*, 2017.

Budi Setiawan. Code-mixing vs code-switching: a study of grammatical perspective through code-switching varieties. *KnE Social Sciences*, pages 47–57, 2023.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steven Bethard, Mona Diab, Mahmoud Ghoneim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, Alison Chang, et al. Overview for the first shared task on language identification in code-switched data. In *Proceedings of the first workshop on computational approaches to code switching*, pages 62–72, 2014.

Lucia Specia, Carolina Scarton, and Gustavo Henrique Paetzold. *Quality estimation for machine translation*. Morgan & Claypool Publishers, 2018.

Vivek Srivastava and Mayank Singh. PHINC: A parallel Hinglish social media code-mixed corpus for machine translation. In Wei Xu, Alan Ritter, Tim Baldwin, and Afshin Rahimi, editors, *Proceedings of the Sixth Workshop on Noisy User-generated*

Text (W-NUT 2020), pages 41–49, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.wnut-1.7. URL https://aclanthology.org/2020.wnut-1.7/.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. Multilingual neural machine translation with knowledge distillation. *arXiv preprint arXiv:1902.10461*, 2019.

Jörg Tiedemann. The Tatoeba Translation Challenge – Realistic data sets for low resource and multilingual MT. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1174–1182, Online, November 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.wmt-1.139.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Haifeng Wang, Hua Wu, Zhongjun He, Liang Huang, and Kenneth Ward Church. Progress in machine translation. *Engineering*, 18:143–153, 2022.

Jitao Xu and François Yvon. Can you traducir this? machine translation for code-switched input. *arXiv preprint arXiv:2105.04846*, 2021.

Zilong Zhong and Lin Fan. Worldwide trend analysis of psycholinguistic research on code switching using bibliometrix r-tool. *SAGE Open*, 13(4):21582440231211657, 2023. doi: 10.1177/21582440231211657. URL https://doi.org/10.1177/21582440231211657.

Appendix A

Simplified Performance Graphs

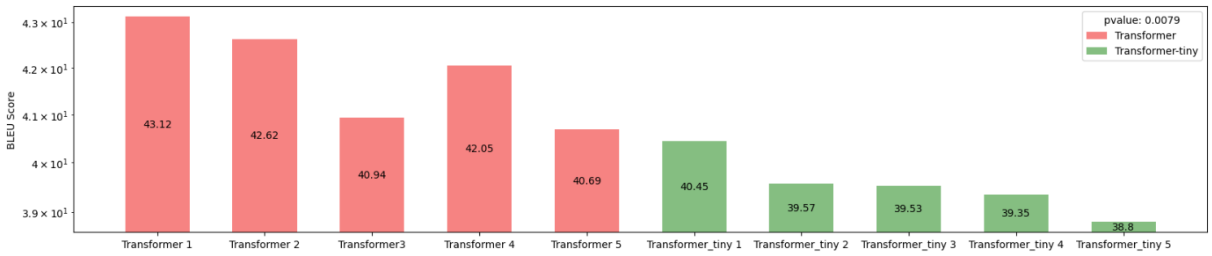


Figure A.1: Baseline 2021 Version - Transformer and transformer-tiny size performance

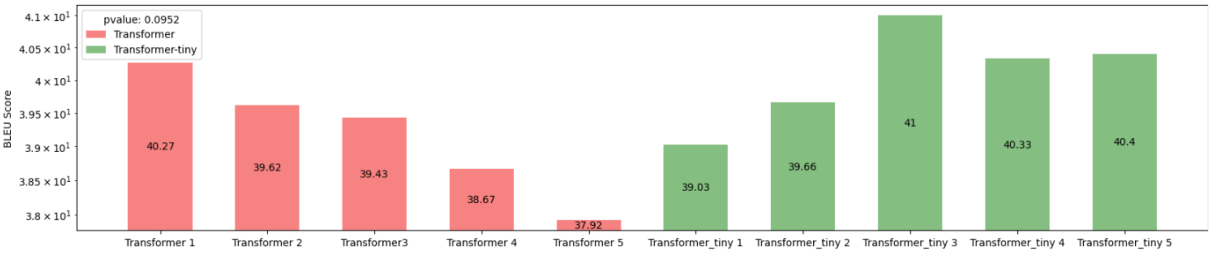


Figure A.2: Baseline 2023 Version - Transformer and transformer-tiny size performance

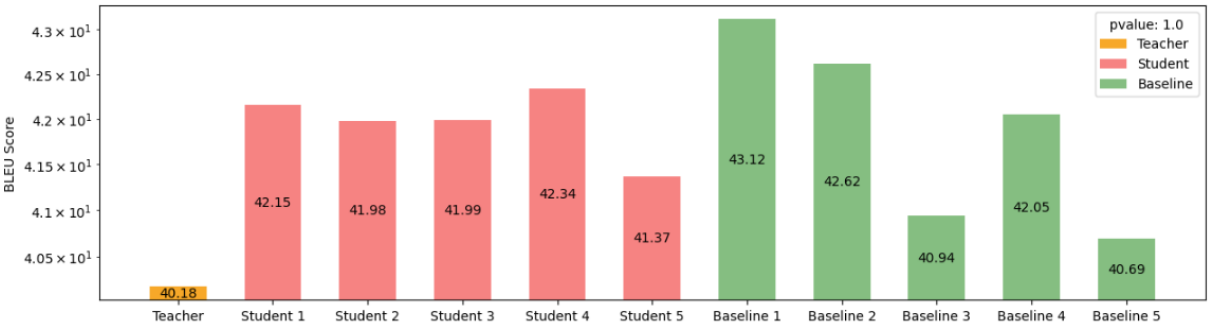


Figure A.3: Model 1 and Baseline 2021 - Transformer size performance

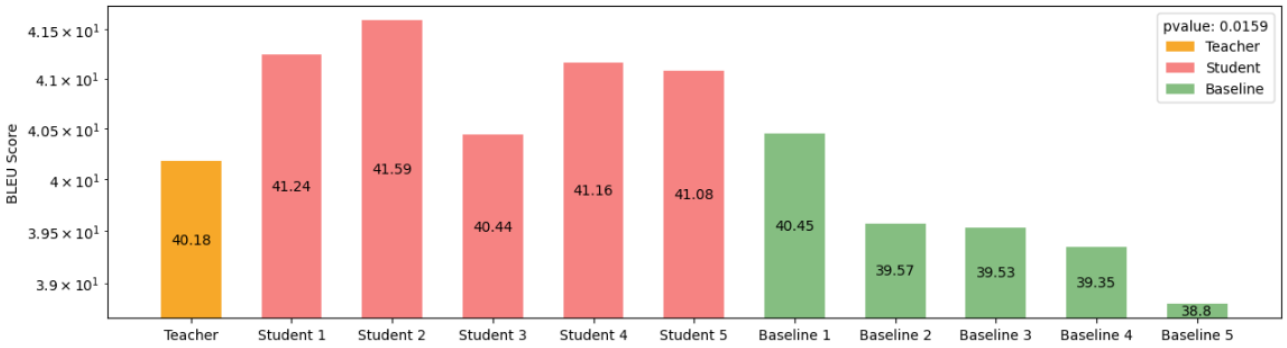


Figure A.4: Model 1 and Baseline 2021 - Transformer-tiny size performance

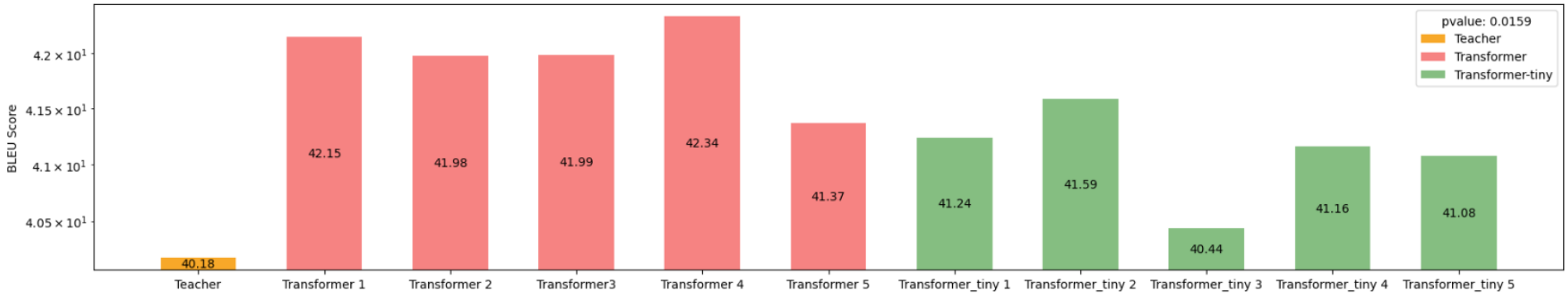


Figure A.5: Model 1 - Transformer and Transformer-tiny size performance

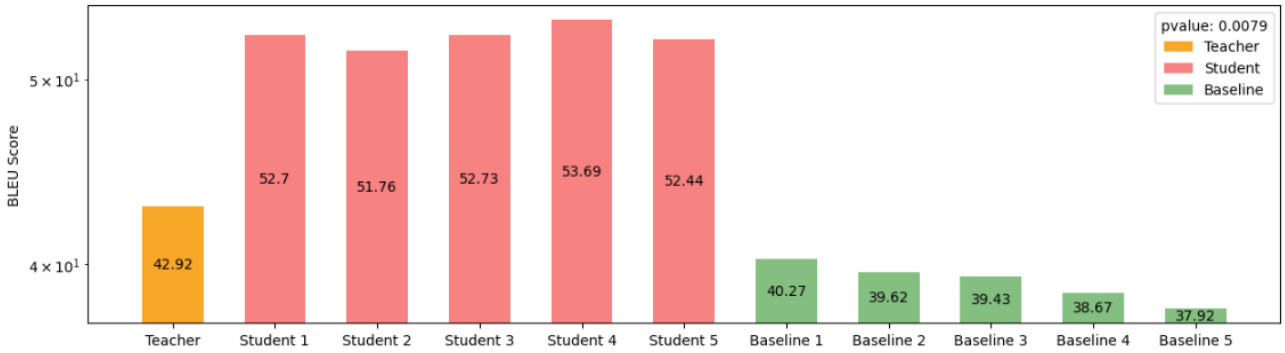


Figure A.6: Model 2 and Baseline 2023 - Transformer size performance

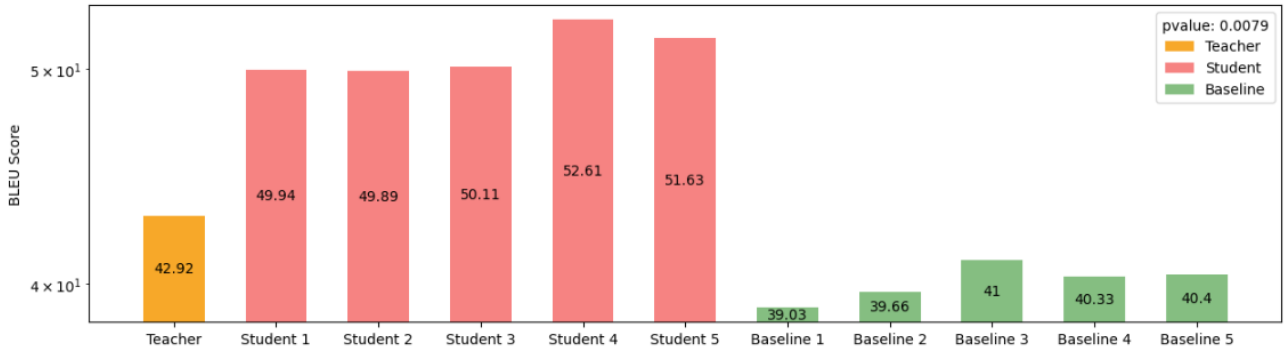


Figure A.7: Model 2 and Baseline 2023 - Transformer-tiny size performance

Figure A.8: Model 2 - Transformer and Transformer-tiny size performance

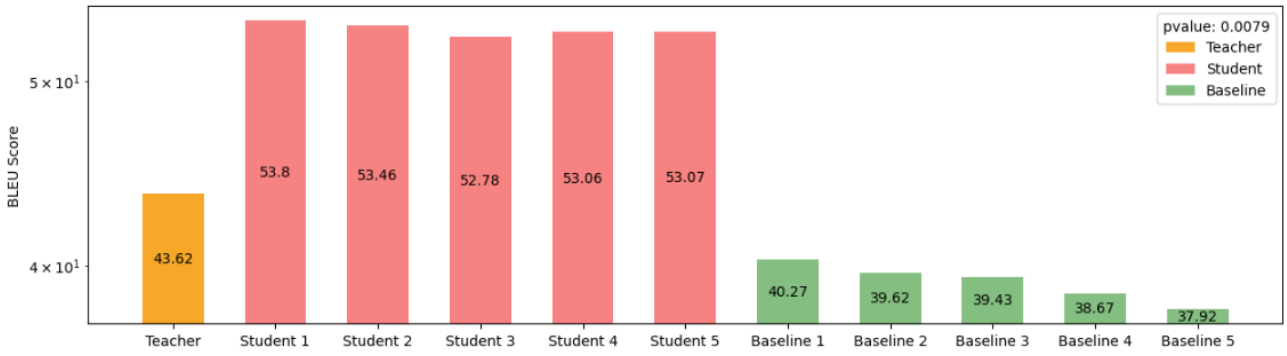


Figure A.9: Model 3 and Baseline 2023 - Transformer size performance

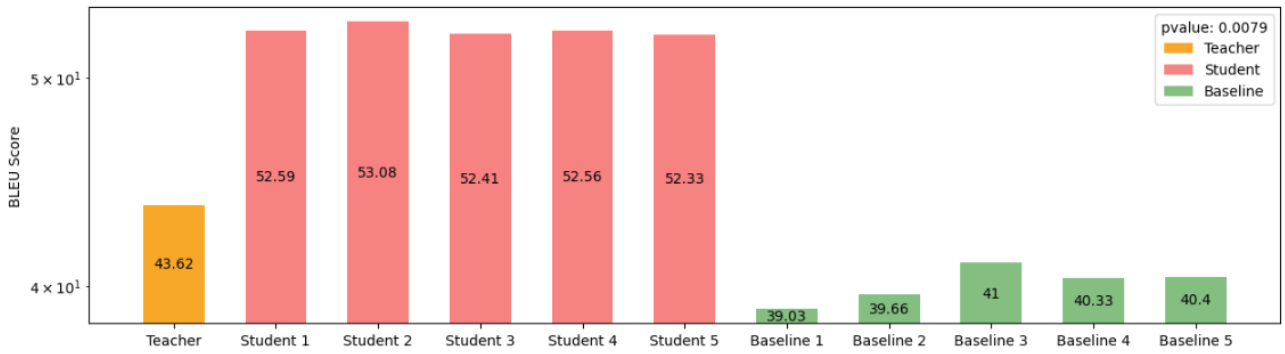


Figure A.10: Model 3 and Baseline 2023 - Transformer-tiny size performance

Figure A.11: Model 3 - Transformer and Transformer-tiny size performance

Appendix B

Full performance Graphs

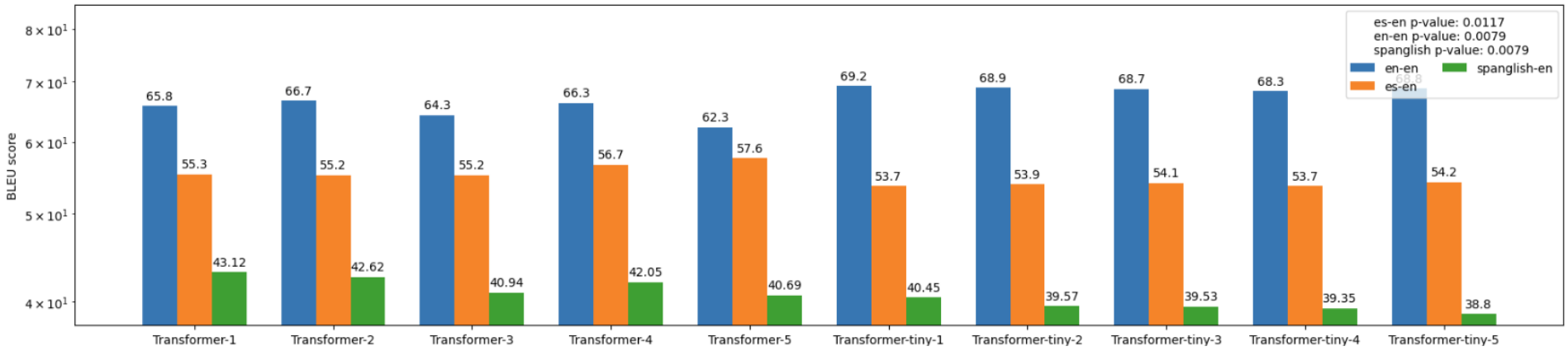


Figure B.1: Baseline 2021 Version - Transformer and transformer-tiny size performance

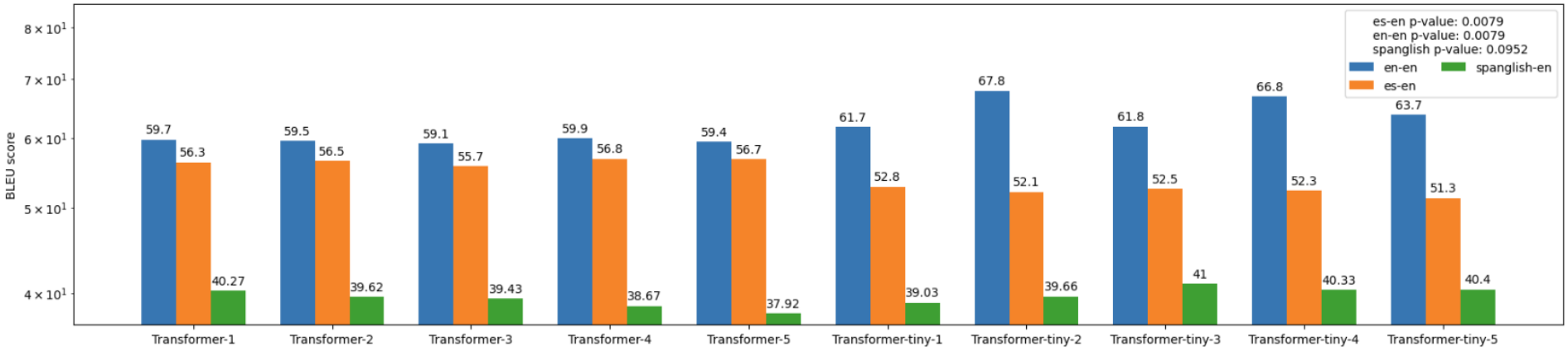


Figure B.2: Baseline 2023 Version - Transformer and transformer-tiny size performance

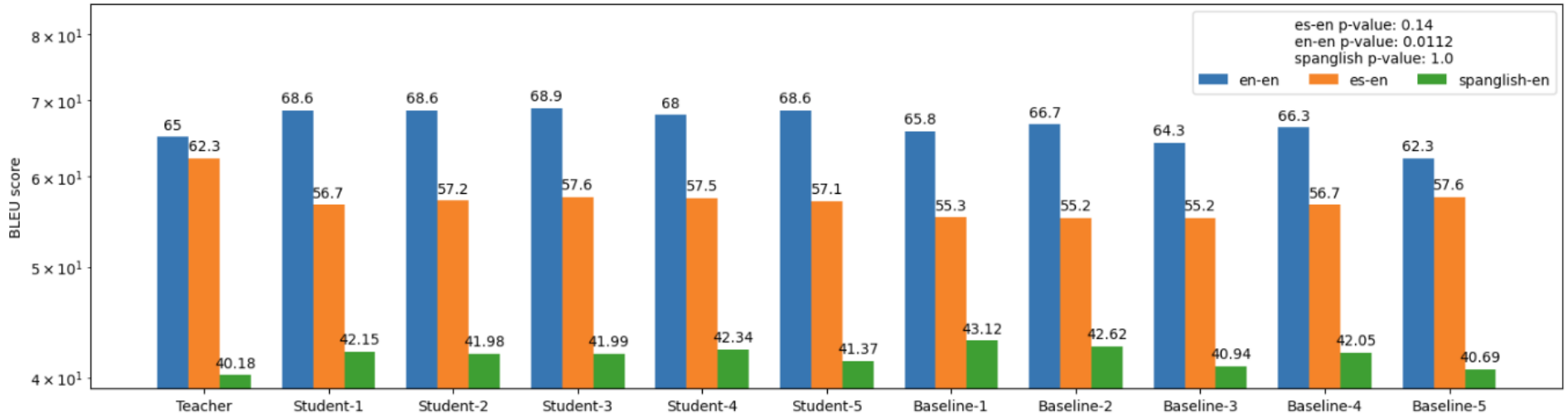


Figure B.3: Model 1 and Baseline 2021 - Transformer size performance

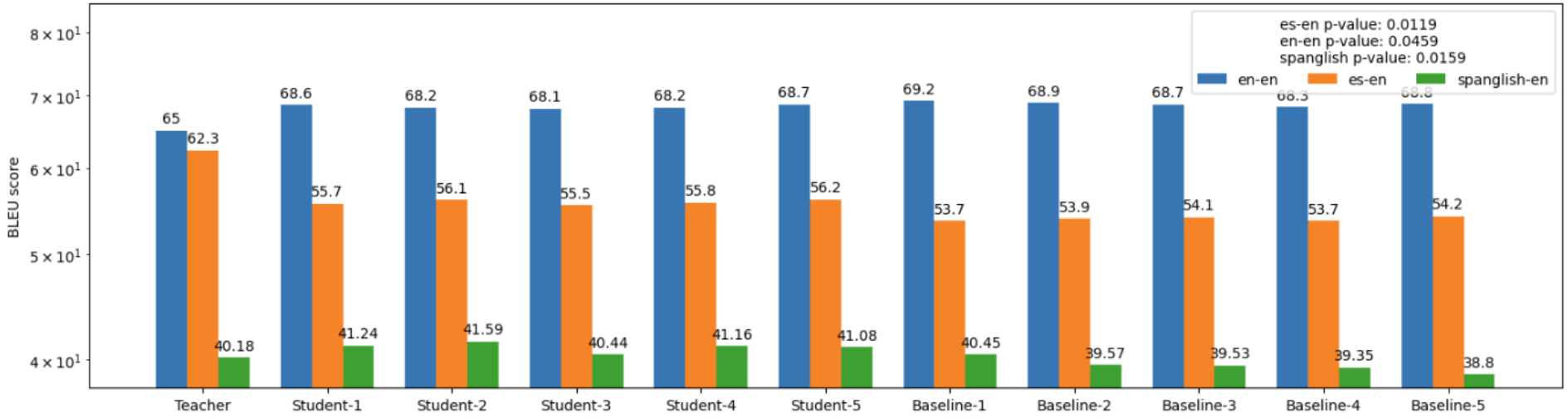


Figure B.4: Model 1 and Baseline 2021 - Transformer-tiny size performance

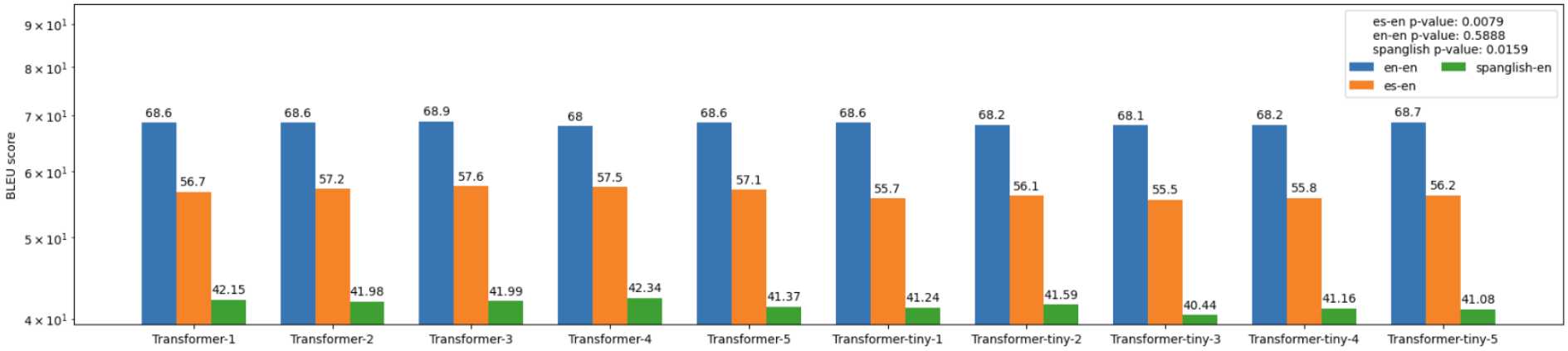


Figure B.5: Model 1 - Transformer and Transformer-tiny size performance

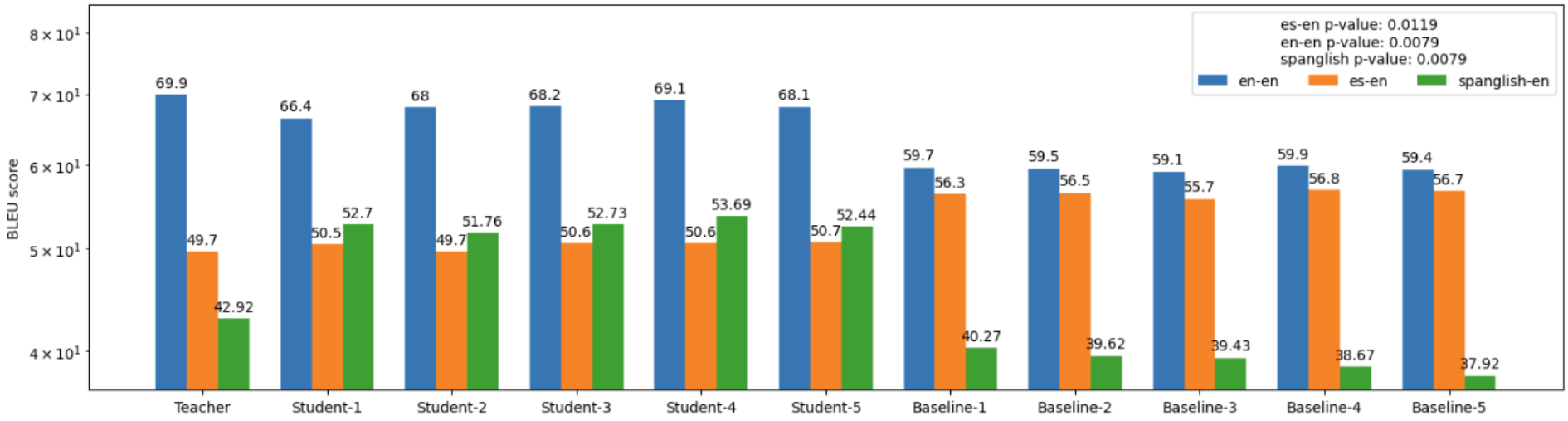


Figure B.6: Model 2 and Baseline 2023 - Transformer size performance

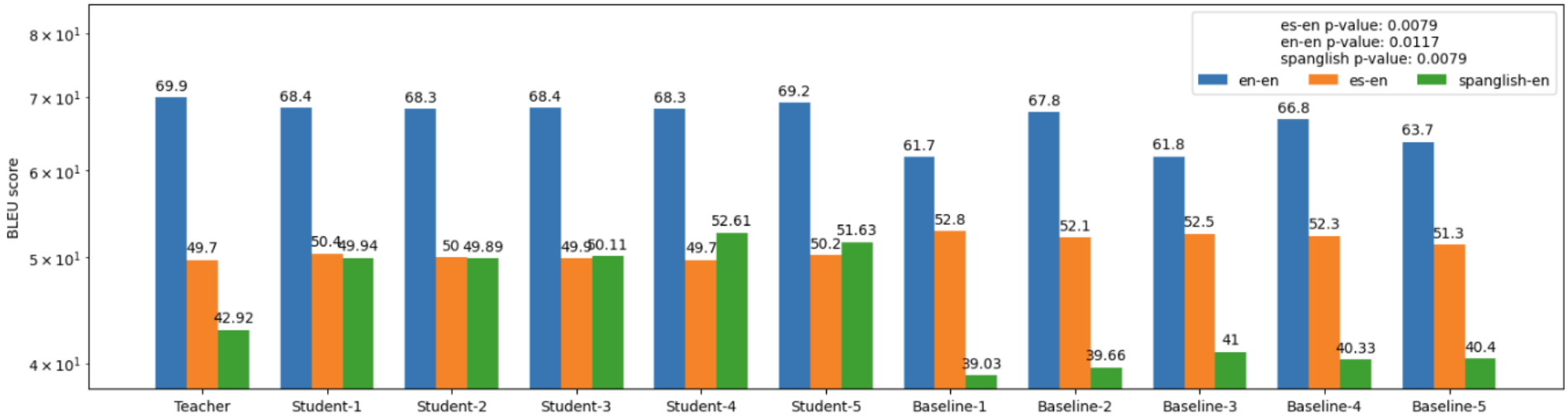


Figure B.7: Model 2 and Baseline 2023 - Transformer-tiny size performance

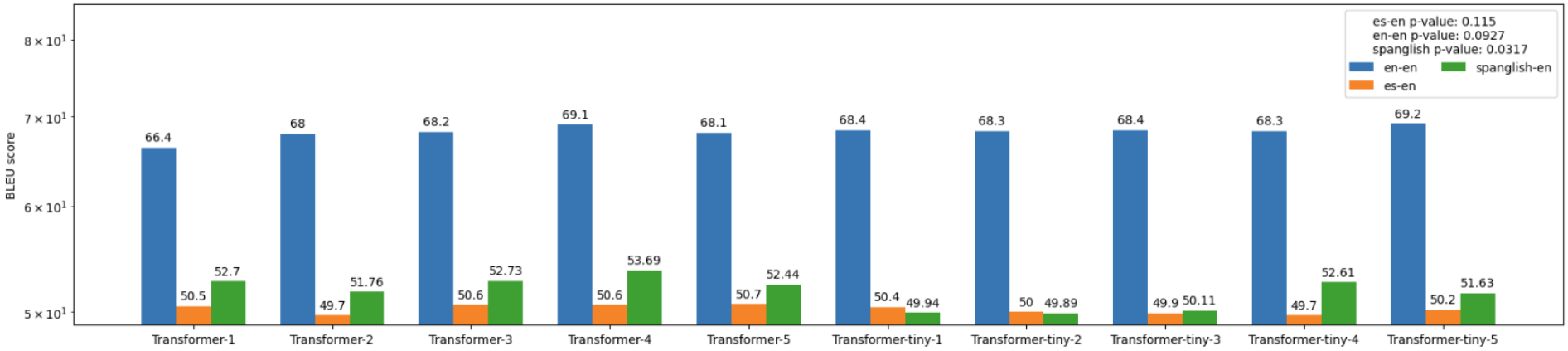


Figure B.8: Model 2 - Transformer and Transformer-tiny size performance

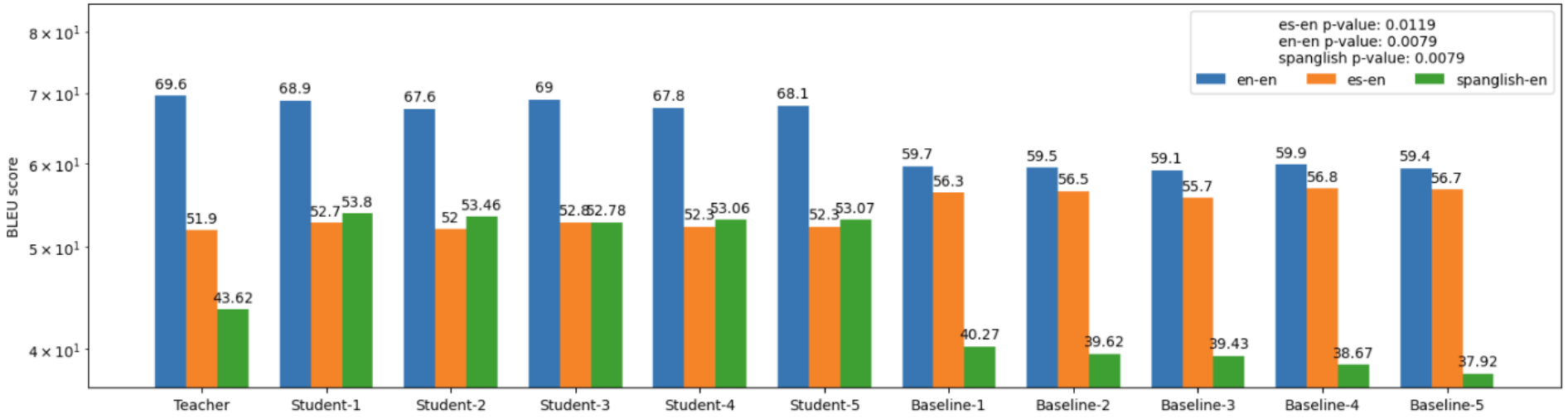


Figure B.9: Model 3 and Baseline 2023 - Transformer size performance

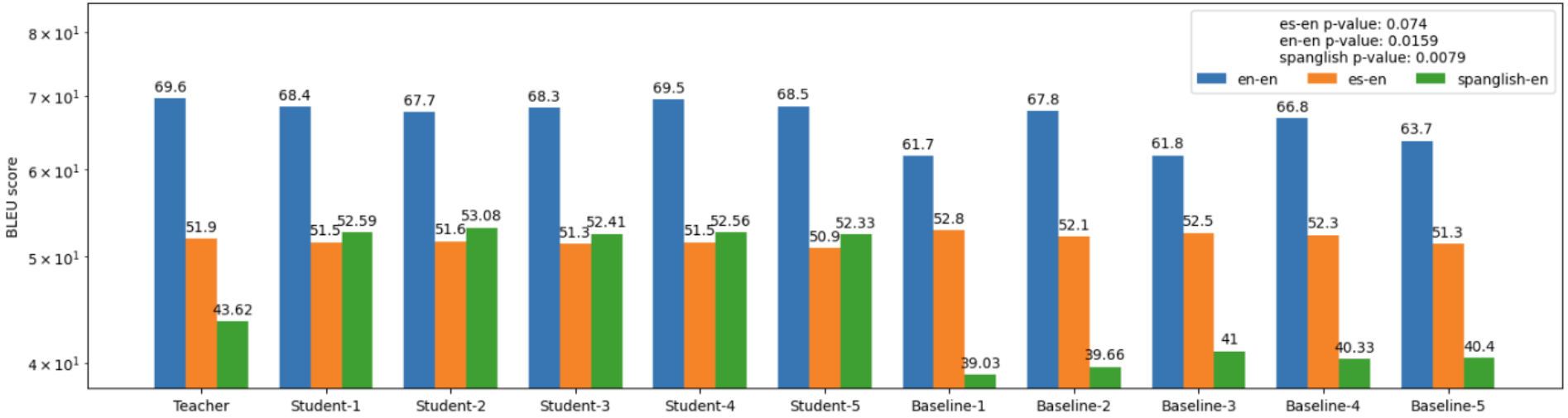


Figure B.10: Model 3 and Baseline 2023 - Transformer-tiny size performance

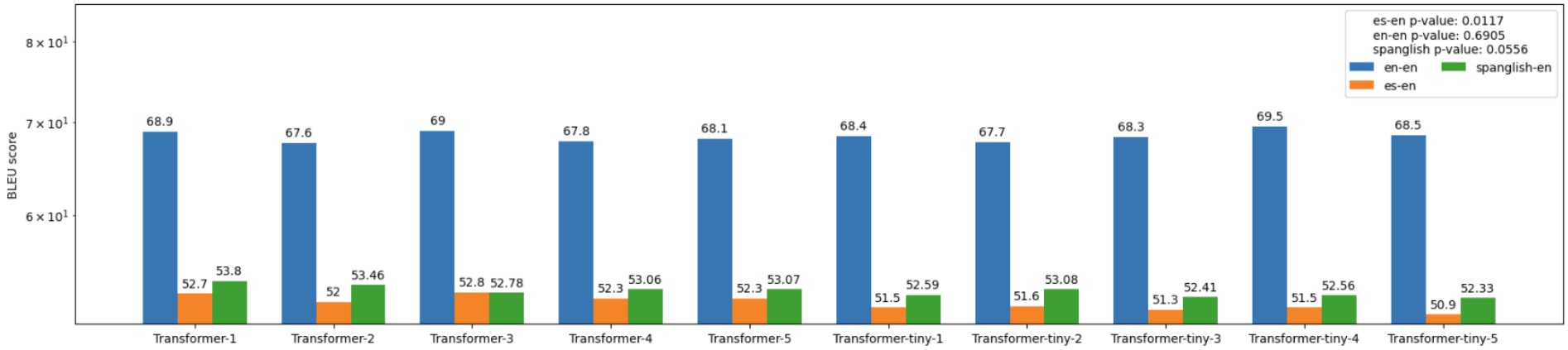


Figure B.11: Model 3 - Transformer and Transformer-tiny size performance

Appendix C

Training and Translation times

Model size	Model type	Model 1	Model 2	Model 3	Model 4	Model 5
Transformer	Baseline 2021	31.72608	23.51835	23.68703	24.78479	26.48066
	Model 1	22.70379	23.10043	16.86551	18.65730	23.37144
	Baseline 2023	32.36374	23.94228	23.19770	24.12650	25.14696
	Model 2	20.85693	20.94667	22.29849	16.40979	24.83564
	Model 3	23.57824	20.65143	21.42397	20.94951	21.77517
Tiny	Baseline 2021	13.78603	8.70405	9.26018	9.11017	10.14164
	Model 1	14.91458	7.25277	9.81955	9.69972	9.22702
	Baseline 2023	8.69276	8.24044	9.01253	9.81285	9.11734
	Model 2	9.94232	10.11225	9.69609	9.36607	9.85835
	Model 3	8.27391	9.73568	9.16993	9.29794	8.67915

Table C.1: Inference times for Student and Baseline models in seconds

Model	Size	Training time
Model 1	Transformer	16h 29min 42sec
		14h 06min 02sec
		19h 34min 10sec
		21h 08min 55sec
		15h 37min 29sec
	Tiny	17h 13min 42sec
		15h 48min 40sec
		15h 53min
		13h 51min 25sec
		17h 59min 15sec
Model 2	Transformer	16h 21min 06sec
		14h 13min 34sec
		19h 06min 37sec
		22h 18min 55sec
		18h 43min 07sec
	Tiny	20h 17min 21sec
		13h 33min 47sec
		15h 05min 16sec
		12h 47min 55sec
		18h 47min 20sec
Model 3	Transformer	21h 23min 20sec
		16h 05min 56sec
		20h 56min 13sec
		16h 12min 59sec
		18h 38min 07sec
	Tiny	14h 06min 21sec
		17h 46min 50sec
		13h 26min 06sec
		14h 08min 35sec
		14h 51min 58sec

Table C.2: Training Times for Student Models

Model	Size	Training time
Baseline 2021	Transformer	14h 42min 21sec
		20h 50min 07sec
		20h 28min 30sec
		39h 07min 18sec
		20h 57min 30sec
	Tiny	17h 27min 43sec
		26h 13min 36sec
		24h 38min 33sec
		21h 03min 29sec
		32h 37min 42sec
Baseline 2023	Transformer	34h 51min 59sec
		31h 00min 32sec
		22h 12min 39sec
		40h 17min 40sec
		58h 15min 27sec
	Tiny	24h 33min 13sec
		20h 23min 53sec
		18h 02min 34sec
		21h 00min 21sec
		13h 21min 46sec

Table C.3: Baseline Models Training time