Enriching Sentence-Level Machine Translation

Proyag Pal

Doctor of Philosophy

THE UNIVERSITY OF EDINBURGH

2024

Abstract

Neural Machine Translation (MT) has long been established as a successful paradigm to produce high-quality MT across many languages and domains. However, it suffers from one significant limitation – it is too often formulated as a task of translating isolated sentences in a source language into sentences in the target language. This renders standard MT models unable to capture any information that is not in the sentence, such as document context, speaker information, the domain of the text, external constraints etc. This thesis aims to study this limitation, analyse the shortcomings of sentence-level MT, and present some approaches to enrich MT models to overcome this limitation.

The first part of this thesis introduces a method to quantify the amount of information missing from source sentences that is needed to translate them perfectly. This method is called "cheat codes" and it allows us to establish an upper bound on the amount of additional information that the model needs to be provided to be able to exactly reproduce reference translations. We find that a surprisingly small amount of leaked information about the target in addition to the source is enough to achieve this. We also use this method to study what parts of translation are difficult for these models to learn correctly, even in the presence of extra information. This analysis allows us to signpost some hard problems for neural MT for further research to focus on.

The second part of the thesis presents two examples of how MT can be augmented with extra information to improve translation quality or overall user experience in specific applications. The first example is using document context, which is always used by human translators when translating text, but is rarely present in parallel corpora. We extract and publish a large-scale dataset of parallel sentences with corresponding contexts from existing publicly available resources, and show that this data helps improve translation performance in terms of overall quality as well as specific document-level phenomena. The second example is providing timing constraints to an isochronous MT model for use in automatic dubbing. By incorporating duration information and keeping track of it while translating, the model can produce translations that better match the source audio, which eventually results in a better user experience when viewing the automatically dubbed content.

On the whole, we find that even though a relatively small amount of information is missing from sentence-level MT, enriching the models with these small pieces of information can have a significant positive impact on the quality and usefulness of MT systems in a wide variety of situations. We provide detailed analyses, datasets, and methods to build better MT systems and encourage future research in this direction.

Lay Summary

Machine Translation (MT), the task of automatically translating from one language to another, is mostly done using models that translate only one sentence at a time. They cannot use any extra information that is not in the sentence, such as context or information about the speaker, which are commonly used by human translators. This thesis analyses this limitation and presents some ways to add extra information to sentence-level MT models to make them better.

The first part of the thesis introduces a method to measure the amount of additional information needed to translate sentences perfectly. We show that while sentence-level models cannot always produce a perfect translation by themselves, only a small amount of extra information about the target translation is enough to enable them to do so. We also use this method to identify some difficult problems for MT models, by studying the aspects of translation that they struggle to solve even when they are given extra information about the target sentence.

The second part of the thesis presents two examples of how MT can benefit from additional information. The first example is using document context, which is known to be necessary for human translation, but is usually absent from datasets used to train MT models. We automatically extract data with document context from existing resources, publish these datasets for future research, and show how adding context to the otherwise sentence-level model improves translation quality. The second example is to use timing information from videos to produce more suitable translations for automatic dubbing. We show that enabling the MT model to keep track of the duration of the source audio while translating improves the quality of dubbed videos.

On the whole, this thesis shows that adding relatively small pieces of extra information to sentence-level MT models can have a big positive impact on the quality and usefulness of MT systems in a wide variety of situations.

Acknowledgements

Getting a PhD is hard. It's harder when you start in the middle of a deadly pandemic and don't see anyone else for the first two years. It might have been too hard without the love and support of a lot of people.

Kenneth Heafield was the person who got me into MT many years ago during my masters, he supported me and gave me opportunities to work on cool things many times over the years, and he reached out to me to come back and do a PhD when I was having a rough time. I wouldn't be writing this thesis without him and I will always be grateful for his mentorship, support, and the brainstorming sessions to generate many of the ideas that are in the next hundred or so pages.

I could not imagine anyone better than Lexi Birch to take over my supervision when Kenneth left. I am grateful for her feedback in making this thesis a lot better and her support and advice over the last year. I can only be sorry that I didn't get to work with her more during my PhD, but I am so excited that I get to keep doing that afterwards.

My thanks to my examiners, John DeNero and Barry Haddow, for taking the time to read this thesis. Suggestions and advice I received from Peter Bell and Shay Cohen at annual reviews were also very useful in shaping the direction of this thesis and keeping me on track.

I'm lucky to have worked with some other brilliant people during this PhD. Brian Thompson and the Doubtfire team at Amazon taught me about the fascinating world of automatic dubbing – it was a huge learning experience, and went on to form an important part of this thesis. Rico Sennrich gave me a lot of amazing insight into my own work while I had the chance to spend some time with his group in Zurich. The StatMT group at the University of Edinburgh is full of great people, and seeing their work and discussing MT with them inspired me. Special mention to the subset of that group that I shared an office and a supervisor with – Laurie and Patrick; people who know exactly what I'm going through are invaluable and helped me keep going.

It's hard to be in a fun headspace when you're trying to figure out how to do a PhD while in lockdown in a pandemic, and my friends Pathok, Sumon, and KB made sure I also had a lot of fun with gaming, watching football, and chatting in the dead of night to stay mentally healthy.

A loving, caring, and supportive family makes it so much easier to embark on a journey as long and lonely as a PhD. My parents are the reason I thought it was worth getting a PhD in the first place. My sister Shoili has always been a source of fun, care, and support. My grandfather is one of the main reasons I was ever able to get an education as good as I have, and I wish he got to see me finish this.

And finally, my wife Evelyn – she is the reason I was able to finish this PhD. Her sacrifices and unwavering support gave me the perfect environment and motivation to give my best and succeed. Problems in my research or a bad paper review turn out to be insignificant when I zoom out and focus on my family and the rest of my life.

This thesis is for all these people. Thank you.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Proyag Pal

Contents

Appendices

Figures and Tables

Figures

Tables

Acronyms

ASR Automatic Speech Recognition. 78, 79

BPE Byte Pair Encoding. 85

CAT Computer-Assisted Translation. 65, 70

GRU Gated Recurrent Unit. 13, 14, 26, 44

LLM Large Language Model. 21, 22, 62, 97

LRP Layerwise Relevance Propagation. 25

LSTM Long Short-Term Memory. 13, 14

MFA Montreal Forced Aligner. 84

MOS Mean Opinion Score. 92

MQM Multidimensional Quality Metrics. 42

MT Machine Translation. ii–v, viii, 1–14, 18–25, 30, 37–43, 47, 49, 57–61, 65, 67, 70, 74–81, 84, 86, 87, 93–97

NER Named Entity Recognition. 47

NLP Natural Language Processing. 2, 18, 21, 37, 97

PA Prosodic Alignment. 78–81

PBMT Phrase-Based Machine Translation. 1

PoS Part of Speech. ix, xi, 20, 40, 47, 49, 51, 56–58, 94

RNN Recurrent Neural Network. 13

TTS Text-to-Speech. 10, 78, 81, 97

Chapter 1

Introduction

The task of Machine Translation (MT), throughout the modern era of Neural MT and the previously prevalent paradigm of Phrase-Based Machine Translation (PBMT), has largely been approached as a sentence-level task. It is formulated as a problem of translating text from a source language to a target language one sentence at a time. This approach usually fails to capture any context, other external information, or user expectations. This effectively assumes that it is in fact possible to faithfully translate each sentence in isolation into another language. However, in reality, this is rarely an unambiguous task.

When a human translator performs the task of translation, they are never given a single isolated source sentence to translate. With access to various other pieces of information – such as context, domain, expected style, etc. (see Section 1.1 for more examples), the translator makes numerous choices about how to translate the provided sentence. This additional information is often necessary for producing a translation that is not only correct but also appropriate for the context and situation in which it is used, and is almost always available to a translator.

MT systems, in their standard form, are typically incapable of incorporating these extra-sentential factors. Historically, this has been due to training corpora being created almost exclusively at the sentence-level and not providing any extra context, due to the fact that richer parallel corpora are much more difficult to obtain in the quantities required to train these models. As a result, MT models were also mostly designed to operate on such data. Most models before recent architectural developments were also limited in terms of the amount of information they could encode beyond a short span of text, and translating larger spans was computationally prohibitive. These limitations led to the field of MT focusing narrowly on improving sentence-level translation, causing the field to remain entrenched in this paradigm for a long time. In recent years, there has been a significant amount of work on document-level MT (see Chapter 5), but a large amount of work in MT continues to be purely sentence-level.

Many other tasks in the broader field of Natural Language Processing (NLP) also benefit from being provided information beyond the basic source text. For example, open-ended language generation models produce more coherent text when they are able to capture more context (Guan et al., 2021), dialogue models can participate in more engaging and consistent conversation when they have access to previous conversational turns or information about the user (Cho et al., 2022; Quan and Xiong, 2020), question answering models can answer questions more accurately given access to knowledge bases (Lewis et al., 2020), etc.

Among all these tasks, machine translation is a task that is particularly well-suited to exploring the effects of incorporating additional information into models. This is because in addition to the task being of great practical application, it has a long history of research and a large active research community, leading to the existence of massive datasets and well-defined and rigorous evaluation metrics and frameworks. This makes it an ideal testbed for studying the advantages of adding various types of extra information to sentence-level models by enriching them with data and mechanisms that enable them to use this information and ultimately produce more useful translations.

Recent advances in neural networks for NLP have enabled the development of models with the capacity to encode much larger amounts of information, including longer context and multimodal input. It has also become standard practice to train language models to accept input far beyond the sentence-level. Models have been built to handle extremely long contexts, but they are rarely evaluated on real-world scenarios that actually require this much information. They are usually not rigorously evaluated on exactly how they use this long context to produce their output, and their benefits for MT have not been studied in depth.

This makes it a good time to revisit and reconsider how the field approaches the task of neural MT – to consider what kinds of information we should be trying to incorporate into MT models and which of them are more difficult to learn, to question whether model capacity is a limitation at all, to create datasets and methods to enable models to use this information, and to evaluate the effects of incorporating this information on translation quality.

1.1 What is Missing from Sentence-Level MT?

Consider the following English sentence:

```
You are beautiful.
```

While it appears to be a fairly simple sentence in English, let's consider how it might be translated into French. If a human were asked to translate this sentence, knowing that the translation is ambiguous, they would most likely first ask for context, or more information about the addressee. The translation of each of the three words in the sentence is ambiguous without additional information.

At the time of writing, Google Translate[1], probably the best-known MT system, translates this sentence as:

```
Tu es belle.
```

ChatGPT[2], also a powerful translation tool, given the following prompt:

```
Translate the following sentence from English to French:
You are beautiful.
```

responds:

```
The translation of "You are beautiful" into French is:
Tu es belle (if speaking to a female)
Tu es beau (if speaking to a male)
```

While this conditional response recognises the inherent ambiguity in the translation, it still does not cover all the possibilities. At the very least, the validity of this translation depends on the grammatical gender, number, and formality of the addressee, as shown in Table 1.1.

This is just one example of how different kinds of information that may be required to produce a certain translation may not be present in the isolated source sentence at all, or even in the entire input text. In some cases, they may be available explicitly, in the form of terminology dictionaries, length constraints, etc. Alternatively, they may be present partially explicitly, such as through translation guidelines or knowledge about the writer/speaker/domain. They may also be entirely implicit, mostly commonly inferred from context, or through expected style and lexical/syntactical choice in case of ambiguity.

[1] https://translate.google.com/?sl=en&tl=fr&op=translate. Translation obtained on 30/09/2024.

[2] https://chatgpt.com/, using the default GPT-4o model, on 24/10/2024.

Gender	Number	Formality	Translation
Feminine	Singular	Informal	`Tu es belle`
Masculine	Singular	Informal	`Tu es beau`
Feminine	Singular	Formal	`Vous êtes belle`
Masculine	Singular	Formal	`Vous êtes beau`
Feminine	Plural		`Vous êtes belles`
Masculine	Plural		`Vous êtes beaux`

Table 1.1: Some possible translations of the English sentence "`You are beautiful`" into French.

Before discussing how this additional information can be obtained or incorporated into MT models, we first discuss some examples of the types of information that may be needed by a model to translate a given sentence accurately and satisfactorily. This list is not meant to be exhaustive, but merely illustrates the variety of additional information that the translation process requires.

Contextual information

The most common form of extra-sentential information comes from surrounding context. Very often, text is translated as part of a longer document, and the preceding and following sentences provide hints to narrow down the translation choices for any given sentence. This is how human translators usually disambiguate cases like the example above. Consider the same sentence, but with the following preceding context:

```
You are my mother. You are beautiful.
```

To a human translator, the correct translation of the second sentence is now relatively unambiguous, since the preceding sentence informs them that the addressee is feminine and singular, and the context is informal. However, a sentence-level MT model without any further modifications, when tasked with translating the latter sentence alone, has no way of using the context from the previous sentence to capture these pieces of information. It may therefore produce an incorrect translation, and a correct translation can be produced only purely by chance. An MT model that is able to leverage information from context should however be able to disambiguate the translation correctly:

```
Tu es belle.
```

There has been a considerable amount of research into incorporating document context into MT models (Jean et al., 2017; Junczys-Dowmunt, 2019; Kuang et al., 2018; Maruf and Haffari, 2018; Miculicich et al., 2018; Post and Junczys-Dowmunt, 2023; Sun et al., 2022; Tiedemann and Scherrer, 2017; Tu et al., 2018; Voita et al., 2018).

However, due to reasons including a scarcity of document-level parallel data and the additional computational cost of processing context, sentence-level models remain widely used in industry and research despite their obvious shortcomings relative to their document-level counterparts. This is discussed in greater depth in Chapter 5.

Length or timing constraints

For some specific applications of MT, the choice of translation may be dictated by external constraints. For example, in an automatic dubbing pipeline, the translation needs to be isochronous, i.e. when converted into speech, the sentence should be of a similar duration as the spoken source sentence. Similarly, where the translation needs to fit into the same space as the source sentence in a fixed layout, such as for subtitles, the translation may need to be isometric, i.e. of similar length. For our running example, the translation may need to be:

```
Tu es magnifique.
```

since this is closer in length to the source sentence in terms of the number of syllables and written length. This kind of information cannot be captured from context and needs to be explicitly provided to the model, and the model needs to be able to adapt its translations to satisfy these constraints. There has been some work on isometric and isochronous translation (Federico et al., 2020a,2; Hu et al., 2021; Lakew et al., 2021,2; Saboo and Baumann, 2019; Tam et al., 2022; Virkar et al., 2021,2; Öktem et al., 2019), which will be discussed further in Chapter 6.

Number, gender, and formality

When the target language requires information like grammatical gender, number, formality of the relationship between speaker and addressee (honorifics), etc. that is not present in the source language, it is impossible to produce and inflect the translation correctly without further clarification. While this information can often be inferred from context (which is often absent itself), this is not always the case, and in some translation scenarios, this information may be explicitly required.

Some previous work has explored controlling these factors through gender/formality markers (Nadejde et al., 2022; Niu et al., 2017; Sennrich et al., 2016a; Vanmassenhove et al., 2018). One well-known application of explicitly using gender information to generate gender-specific translations is in Google Translate[3] for some languages such as French and Turkish (Johnson, 2018). However, in most cases, these annotations are not explicitly available during translation, and this remains a common failure mode of MT systems.

We do not explore this kind of information directly in this thesis, but it is a common aspect of translation that is often a limitation of sentence-level MT, and constitutes an important part of the missing information required to improve the accuracy of translations in certain language pairs. Some other such factors are briefly described below, and while we do not directly address these in this thesis, they all illustrate the variety of information that may be used to enrich sentence-level MT.

Terminology

Some MT applications require certain words and phrases, usually specialised vocabulary, to be translated in specific ways. This can be specified as a terminology dictionary consisting of terms and their corresponding translations, which should be used by the MT system to translate the terminology correctly and consistently. While there has been a considerable amount of research into enforcing terminology constraints in MT models (Bergmanis and Pinnis, 2021; Dinu et al., 2019; Hasler et al., 2018; Song et al., 2020; Susanto et al., 2020), it is only used in very specific applications, and is often still unable to generate satisfactory translations while maintaining consistent terminology usage.

Lexical/syntactical choice

Sometimes, even if a default translation is fully correct, a user may expect a different choice of words or sentence structure. While this does not affect the *correctness* of a translation, this is nevertheless information that would be useful for a model to be aware of and potentially be able to incorporate. This kind of information can often be implicitly provided as a domain specification, since some lexical or syntactical choices may be more appropriate for certain domains than others.

[3] `https://translate.google.com/`. This feature is only available for single words and short phrases.

Style and translation guidelines

In a more abstract way, any translation process can include guidelines for the style in which the translations should be done, as well as reflect the personal style of the translator. This information is very difficult to encapsulate in data, but is implicitly encoded into any translation. There has been some work in personalising MT to reflect the style of translators (Rabinovich et al., 2017; Wang et al., 2021b). This can also be implicitly available from context or from the domain; for example, a translation of parliamentary proceedings should use formal language and appropriate terminology.

Multimodal information

In some cases, an image or audio clip associated with text can provide disambiguating information during translation. There is some research on multimodal MT (Caglayan et al., 2016; Lala and Specia, 2018; Yao and Wan, 2020), but this does not correspond to the most common translation scenarios where MT is usually used, and we do not explore this in this thesis.

When we model MT solely as a task of translating a sentence of source words to a sentence of target words, we lose much of this information, resulting in sub-optimal translations that may range from simply being grammatically wrong to subtly failing to meet style specifications while possibly appearing entirely correct out of context.

For years, there have been premature claims of MT achieving “human parity” (Hassan et al., 2018). However, further scrutiny (Läubli et al., 2018; Toral et al., 2018) has shown that this is due to sentence-level systems being evaluated at the sentence-level in restricted domains, and these claims do not hold up in more general settings such as evaluation on full documents or over a variety of domains.

For MT to attain a level of quality that could fully satisfy requirements and expectations, it is therefore important to enrich these models with additional information. This information may simply be necessary to produce a correct translation at the bare minimum, i.e. a sentence-level model would have no way of differentiating the correct translation from incorrect alternatives with the input that it is given. But in cases where a sentence-level model generates a translation that is strictly *correct*, the information may instead be necessary to translate in a way that better conforms to style or domain specifications, to satisfy terminology constraints, or to meet length or timing requirements. This thesis is an exploration of this idea of enriching MT models to improve translation.

1.2 Research Questions and Thesis Contributions

In this thesis, we investigate a few different research questions about enriching sentence-level MT and describe our contributions towards addressing these questions. We divide these questions into two main categories: analysis of the additional information required by MT models; and data and methods to enrich sentence-level MT models with extra information to improve translation.

1.2.1 Analysis of Additional Information Required by MT Models

The first two content chapters in this thesis (Chapter 3 and 4) focus on analysis of the information required by MT models that is not present in the source sentence, and how difficult it is for models to learn to translate certain aspects of sentences accurately despite access to such additional information.

How much information does a model need in addition to the source sentence to produce a desired translation? How can we quantify this?

In Chapter 3, we devise a method to estimate the amount of information missing from source sentences that is required by an MT model to generate expected translations. The method involves leaking a controlled amount of target-side information to the model and measuring the effect of different amounts of leaked information on translation quality. We call this method “cheat codes”, since we essentially allow the model to “cheat” by using target-side information. Using this method, we can estimate the amount of information required by the model to reproduce reference translations.

Quantifying the amount of additional information required to translate accurately is important to understand how limited the sentence-level information usually provided to the models is, how efficiently these models can capture any extra required information, and whether this is a problem of model capacity, training methods, or of training data.

What parts of translation are difficult for models to learn even with additional information about the target available to it?

When we leak increasing amounts of information to the model, we expect that the model should be able to use the additional information to generate more accurate translations. Therefore, when the model uses large “cheat codes”, meaning it has access to a large amount of target-side information, if the model still makes certain mistakes, we can infer that these are difficult problems for the model to translate. In Chapter 4, we thus use the cheat codes method to build models that have access to different amounts of target-side information and analyse their outputs to identify problems that the models appear to find more difficult to learn to solve.

This analysis can help us understand what kinds of information are difficult for neural MT models to learn to encode, and which areas research should focus on to improve the quality of these models.

1.2.2 Enriching MT Models with Additional Information

The final two content chapters in this thesis focus on specific scenarios where models need to use or can benefit from using additional information to produce more accurate or suitable translations.

How can we incorporate additional information into models to improve translation quality and usefulness?

Since intra-document context is the most obvious source of information that is missing from sentence-level models, we explore in Chapter 5 how context information can be automatically extracted from existing datasets and incorporated into MT models to improve translation quality. We first introduce a large-scale parallel corpus with document context to enable document-level machine translation. We then use multi-encoder models to incorporate document context into MT models to improve overall translation quality. The released data and methods can be used to enable further research into document-level MT and hopefully to encourage a shift away from purely sentence-level MT models.

In Chapter 6, we introduce a method to incorporate timing information into MT models, which can be used to generate isochronous translations, i.e. translations that have similar timing to the spoken source sentence, so that the translated output can be used for automatic dubbing. This is an example of how external non-textual information that is not provided in text input can be used to adapt translation to specific situations.

How and how much does adding this information improve translation in specific applications?

In both Chapter 5 and 6, we evaluate the overall effect of incorporating additional information into the sentence-level models in different ways that are relevant to the specific scenarios, and not just in terms of translation quality metrics.

In addition to the standard automatic MT evaluation metrics, Chapter 5 contains detailed evaluations and analyses on targeted discourse phenomena for document-level translation that cannot be modeled at the sentence level, and on the amount of context that is captured by the models in order to model these phenomena.

In Chapter 6, we use both automatic metrics and human evaluation results to show how dubbed videos using isochronous MT models are not just accurately translated, but qualitatively better perceived by users. This shows that explicitly utilising other externally available information can enable an overall improved user experience of automatic dubbing.

1.3 Individual Contributions

Since this thesis is composed of research published in several papers with multiple authors, this section identifies my individual contributions in each paper.

- Pal and Heafield (2022): The idea was jointly designed by both authors of the paper. All experiment design, implementation, analysis, and writing was done by me.
- Pal and Heafield (2023): The idea was a follow-up from Pal and Heafield (2022) devised by both authors of the paper. All implementation, analysis, and writing was done by me.
- Pal et al. (2024): All implementation, analysis, and writing was done by me. Co-authors provided guidance on availability of data sources which were used for this work, and on evaluation datasets.
- Pal et al. (2023): Implementation of our method, evaluations, and analysis were done solely by me. Co-authors provided the idea, implementation of baseline models, modified Text-to-Speech implementations, and datasets.
- Agarwal et al. (2023): I provided baseline models and code while co-organising the automatic dubbing shared task, and participated in human evaluation as a judge. The rest of the work was by co-authors.

1.4 Thesis Structure

The main content of this thesis is organised into four chapters (3 to 6). Each of these chapters presents some work either using information missing from source sentences to analyse MT models or incorporating such information into models to improve some aspect of translation. The remainder of the thesis is thus structured as follows:

Chapter 2 - **Background** presents an overview of background knowledge underlying the content this thesis and a review of some relevant literature.

Chapter 3 - **Cheat Codes to Quantify Missing Source Information** presents "cheat codes" as a method to estimate the amount of information missing in a source sentence that is required to generate a correct translation.

Chapter 4 - **Cheating to Identify Hard Problems for Neural MT** uses the cheat codes method to analyse neural MT models and identify problems that they find difficult to translate accurately.

Chapter 5 - **Document-Level MT with Large-Scale Public Parallel Corpora** describes the creation of large parallel corpora with context and explores how context information can be used to improve translation quality.

Chapter 6 - **Improving Isochronous MT for Automatic Dubbing** presents a method to incorporate timing constraints into models to produce translations better suited for automatic dubbing.

Chapter 7 - **Conclusion** summarises the contributions and findings of the research presented in this thesis and discusses some ideas for future work.

Chapter 2

Background

Machine Translation (MT) is the task of automatically translating text or speech from one language to another. MT is an instance of a category of machine learning problems known as **sequence-to-sequence**. Sequence-to-sequence problems take a sequence of tokens as input and generate another sequence of tokens as output. The input and output sequences can have different and typically variable lengths. This class of problems therefore also includes tasks such as automatic summarisation, paraphrasing, code generation, and many others.

The dominant paradigm for MT over the last many years has been **Neural MT**, where we use a neural network to model the conditional probability $P(Y|X)$ of target sequences $Y = \{y_1, y_2, \ldots, y_{|Y|}\}$ given a source sequence $X = \{x_1, x_2, \ldots, x_{|X|}\}$ as:

$$P(Y|X) = \prod_{t=1}^{|Y|} P(y_t | y_{<t}, X) \tag{2.1}$$

In neural MT, each input sequence X is almost invariably a sentence in the source language, which we represent as a sequence of tokens[1], and the output sentence is a translated sentence in the target language. While some approaches do use longer sequences, such as paragraphs or documents, the standard approach is still to translate one sentence at a time to keep the sequence length manageable. We discuss document-level translation in depth in Chapter 5.

To perform neural MT and many other sequence-to-sequence tasks, the standard neural network architecture used is the **encoder-decoder**. In Section 2.1, we describe the encoder-decoder architecture, which forms the basis of all models used in this thesis.

[1] Tokens can be words, subwords, characters, or even bytes. We have used subword tokens throughout this thesis except where specified otherwise.

The neural network is trained to minimise the negative log likelihood loss over a training corpus D of text pairs (X,Y):

$$\mathscr{L}_{NLL} = -\frac{1}{|D|} \sum_{(X,Y)\in D} \log P(Y|X) \tag{2.2}$$

and at inference, we use beam search to search for the highest probability translation $\hat{Y}$ given an input X:

$$\hat{Y} = \underset{Y}{\operatorname{argmax}}\, P(Y|X) \tag{2.3}$$

After covering the basics of the encoder-decoder neural MT architecture in Section 2.1, we describe some modifications and mechanisms that are used to model extra information in these sequence-to-sequence models in Section 2.2, along with some related work on modelling extra-sentential information.

2.1 Encoder-Decoder Architectures

The encoder-decoder architecture, as the name implies, is composed of two main components: an **encoder** and a **decoder**. The encoder takes the input sequence and generates an intermediate representation, which the decoder then transforms into the output sequence. The encoder-decoder neural network is the underlying architecture for every model used in this thesis, so we describe its structure and components in detail here.

The encoder and the decoder can be implemented using different architectures, which have evolved over time from convolutional networks and Recurrent Neural Networks (RNNs) (Kalchbrenner and Blunsom, 2013) to the Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014) to the Gated Recurrent Unit (GRU) (Cho et al., 2014b) to the Transformer (Vaswani et al., 2017).

A limitation of early encoder-decoder architectures such as those used by Kalchbrenner and Blunsom (2013), Sutskever et al. (2014), and Cho et al. (2014b) was that the encoder generated a fixed-length representation of the input sequence, and this representation would then be decoded to generate the output sequence. This created an information bottleneck, as all the information in the source sequence had to be compressed into a fixed-length vector, which degraded model performance especially for long sequences (Cho et al., 2014a). The solution to this problem was the introduction of the **attention** mechanism by Bahdanau et al. (2015) (see Section 2.1.2).

In the following subsections, we describe the GRU, attention, and Transformers, which form the main components of the encoder-decoder models used in this thesis.

2.1.1 Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU) is a modified and simplified version of the LSTM (Hochreiter and Schmidhuber, 1997). It was introduced in Cho et al. (2014b) and due to its simple architecture and strong performance, was the most commonly used encoder and decoder architecture in MT models for several years. While the GRU is no longer commonly used and has largely been superseded by Transformers (see Section 2.1.3), we present it here since we have used GRUs as additional encoders in Chapter 3 and Chapter 4.

A GRU maintains an internal hidden state $\mathbf{h}$, which is updated at each sequence step. At step t, the GRU computes the reset gate $\mathbf{r}_t$ and the update gate $\mathbf{z}_t$ as:

$$\mathbf{r}_t = \sigma(W_{rx}\mathbf{x}_t + W_{rh}\mathbf{h}_{t-1}) \tag{2.4}$$

$$\mathbf{z}_t = \sigma(W_{zx}\mathbf{x}_t + W_{zh}\mathbf{h}_{t-1}) \tag{2.5}$$

where $W_{rx}, W_{rh}, W_{zx}, W_{zh}$ are learned weight matrices[2], $\mathbf{x}_t$ is the embedded input at time step t, and $\mathbf{h}_{t-1}$ is the previous hidden state. The hidden state $\mathbf{h}_t$ is then updated as:

$$\tilde{\mathbf{h}}_t = \tanh(W_{hx}\mathbf{x}_t + W_{hh}(\mathbf{r}_t \odot \mathbf{h}_{t-1})) \tag{2.6}$$

$$\mathbf{h}_t = \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \tilde{\mathbf{h}}_t \tag{2.7}$$

When used as an encoder in an encoder-decoder model without attention, the final hidden state $\mathbf{h}_{|X|}$ is used as the fixed-length representation of the input sequence. It is also common practice to have a second GRU encode the sequence in reverse order, and use the concatenation of the forward and backward encoders as the input encoding. This is known as a bidirectional encoder, and we use a bidirectional GRU wherever we use a GRU as an encoder in this thesis.

[2] We omit bias terms with the matrix multiplications for simplicity.

When used as a decoder, the output tokens y_t are generated using a linear projection from the hidden state $\mathbf{h}_t$ followed by a softmax layer and mapping the output probabilities to the output token vocabulary.

2.1.2 Attention

Bahdanau et al. (2015) added the attention mechanism to the encoder-decoder architecture, which enabled the decoder to focus selectively on parts of the source sequence representations instead of relying on a fixed-length source representation. At each decoder step, the attention mechanism creates a context vector as a weighted sum of the encoder states, where the weights are a measure of the relevance of each encoder state to the current decoder state. At sequence step t, given encoder states $\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{|X|}\}$ and decoder state $\mathbf{s}_{t-1}$, the context vector $\mathbf{c}_t$ is computed as:

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{|X|} \exp(e_{t,j})} \tag{2.8}$$

$$\mathbf{c}_t = \sum_{i=1}^{|X|} \alpha_{t,i} \mathbf{h}_i \tag{2.9}$$

where $\alpha_{t,i}$ is the attention weight computed for encoder state i at decoder step t. The scores $e_{t,i}$ between the decoder and encoder states can be computed in different ways – we use **scaled dot-product attention** (Vaswani et al., 2017) in all our models (see Section 2.1.3). The decoder uses the context vector $\mathbf{c}_t$ along with its own current state $\mathbf{s}_{t-1}$ to update its state $\mathbf{s}_t$ and generate the next output token y_t. The model can therefore attend to only a small part of the input sequence at a time instead of having to condition on the entire encoded input.

Luong et al. (2015a) introduced modified attention mechanisms that further improved performance. Vaswani et al. (2017) then introduced the **Transformer** architecture, where attention operations are used as the central mechanism to model the dependencies between tokens in both the encoder and decoder as well (see Section 2.1.3).

2.1.3 Transformer

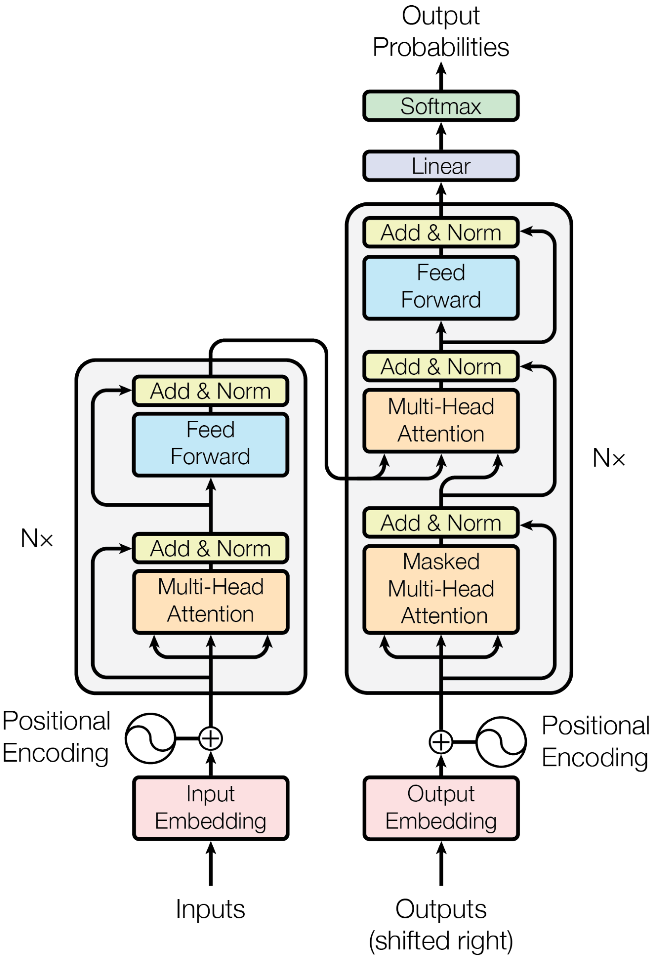

Figure 2.1: The encoder-decoder Transformer architecture. Image from Vaswani et al. (2017).

The Transformer architecture (Vaswani et al., 2017) uses attention as the main building block for both the encoder and decoder. Both encoder and decoder are composed of a stack of identical layers, and each layer takes the output of the previous layer as input (the first layer takes the embedded sequences as input) and transforms it. Figure 2.1 shows the components of the Transformer architecture, and the main individual components are described below.

Scaled Dot-Product Attention

The attention mechanism used in the Transformer and throughout this thesis is known as **scaled dot-product attention**. Given queries Q, keys K, and values V, the attention weights are computed as:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.10}$$

When the queries, keys, and values come from the same source, it is called **self-attention**, and this is used to transform sequence representations by attending to different parts of the same sequence to capture internal dependencies and relationships. When the queries come from the decoder and the keys and values come from the encoder, it is called **cross-attention**, and this is used to allow the decoder to attend to the encoded information when generating the output sequence.

The Transformer uses a modification of the scaled dot-product attention called **multi-head attention**, where the queries, keys, and values are linearly projected h times (each called a *head*) and the attention is computed separately for each head. The outputs of the heads are concatenated and linearly projected again to get the final output. This allows the model to attend to different parts of the input simultaneously and capture more complex interactions.

Encoder

Each encoder layer has two sublayers: a self-attention mechanism and a feed-forward sublayer. The self-attention sublayer attends over the input **x** and outputs a new contextualised representation. The feed-forward sublayer is then applied position-wise to the output of the self-attention sublayer. The output of each encoder layer becomes the input of the next layer.

$$\text{EncoderLayer}(\mathbf{x}) = \text{FeedForward}(\text{SelfAttention}(\mathbf{x})) \tag{2.11}$$

Decoder

In addition to the self-attention and feed-forward sublayers, the decoder layers have an additional cross-attention sublayer between the two. This sublayer attends to the encoder output, allowing the decoder to focus on different parts of the input sequence. In the decoder, the self-attention is masked so that the model cannot attend to tokens in positions to the right of the current token.

$$\text{DecoderLayer}(\mathbf{y}, \mathbf{x}) = \text{FeedForward}(\text{CrossAttention}(\mathbf{x}, (\text{SelfAttention}(\mathbf{y}))) \quad (2.12)$$

We omit some details of the Transformer architecture, such as positional encodings, layer normalisation (Ba et al., 2016), and residual connections (He et al., 2016), since they are not explicitly referenced or modified in this thesis. These are shown in Figure 2.1 and we refer the reader to the original paper (Vaswani et al., 2017) for a full description of the architecture.

The Transformer architecture has established itself as almost ubiquitous, not just in MT but in all of NLP, and even in other domains such as computer vision (Dosovitskiy et al., 2021). All the models used in this thesis are based on the Transformer architecture, with some modifications to incorporate extra information, as described in Section 2.2.

2.2 Modelling Extra Information

In this section, we describe a few mechanisms that are used to inject extra information into MT models, mostly focusing on the encoder-decoder neural MT architecture. This is far from an exhaustive list, but covers some of the most common methods used in the literature, focusing on those used in this thesis. In the process, we also survey some related work that has used these methods to incorporate specific types of additional information into sentence-level MT models.

2.2.1 Multi-Encoder Architectures

The encoder-decoder model can be extended to have multiple encoders, each taking different inputs and encoding different information. The decoder can attend to each encoder separately and combine the information from all of them.

Zoph and Knight (2016) presented a method to use multiple encoders to provide input in multiple languages to MT models to improve translation quality. Dual encoder networks have been used in language generation tasks to inject conversational context into dialogue models (S. et al., 2017), encode input at different levels of granularity (Yao et al., 2020), or for context awareness (Jean et al., 2017; Li et al., 2020; Voita et al., 2018). Junczys-Dowmunt and Grundkiewicz (2017) and Junczys-Dowmunt and Grundkiewicz (2018) use dual-encoder architectures for automatic post-editing, and the second input in that case is MT output to be post-edited.

We use the modified multi-encoder Transformer architecture from Junczys-Dowmunt and Grundkiewicz (2018) in Chapter 3, Chapter 4, and Chapter 5, adding a second encoder to incorporate additional inputs into the encoder-decoder model. Each decoder layer in the dual-encoder Transformer has two stacked cross-attention sublayers, each attending to a different encoder. This architecture is shown in Figure 2.2.

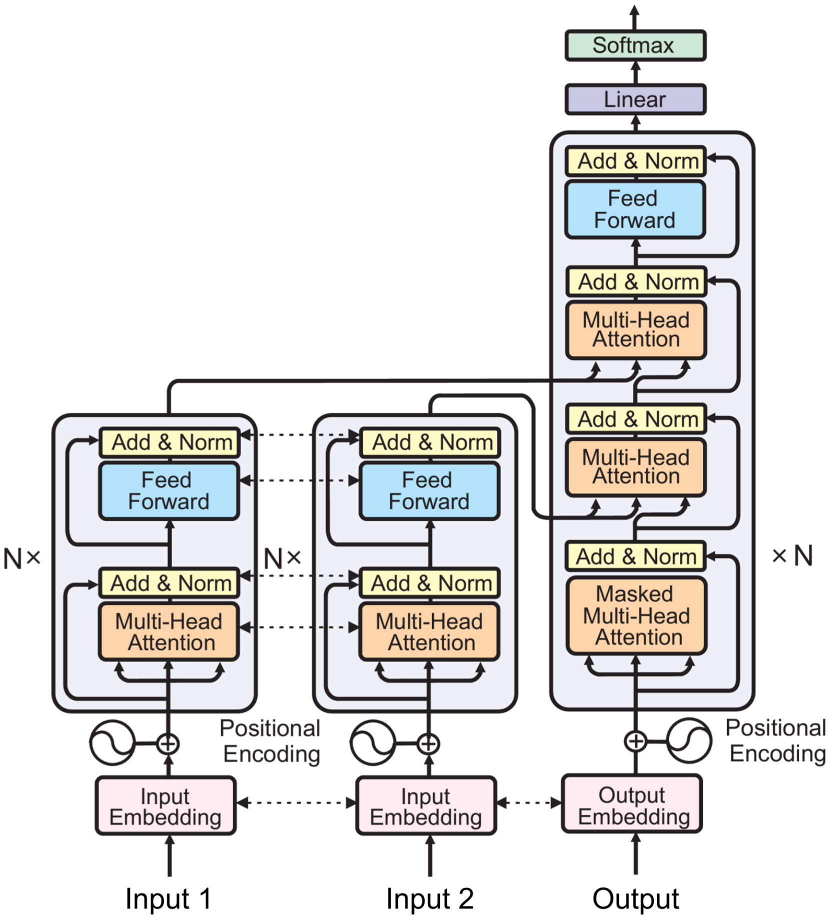

Figure 2.2: Dual-encoder Transformer architecture. Image from Junczys-Dowmunt and Grundkiewicz (2018).

There are more complicated architectures used to input context in document-level MT (Kuang et al., 2018; Maruf and Haffari, 2018; Miculicich et al., 2018; Tu et al., 2018), for example, but in this thesis, the only architecture we use beyond the standard encoder-decoder is the dual-encoder variant.

2.2.2 Pseudo-Tokens

Pseudo-tokens, also referred to as tags, are special tokens that are appended or prepended to the input sequence to convey some information in addition to the input tokens. Commonly used to specify the target language in multilingual MT models (Aharoni et al., 2019; Johnson et al., 2017; Wicks and Duh, 2022), pseudo-tokens are also used to add various kinds of information to guide the output.

For example, Sennrich et al. (2016a) and Nadejde et al. (2022) used a pseudo-token `<T>` or `<V>` to specify politeness (Brown and Gilman, 1968) when translating English to German, since German uses honorifics while English does not. As shown in the example below, the pseudo-token does not form part of the actual sentence being translated, but provides guidance to the model on how to generate the translation.

```
Give me the telephone! <V> → Geben Sie mir das Telefon!
Give me the telephone! <T> → Gib mir das Telefon!
```

Similarly, Vanmassenhove et al. (2018) prepended a `FEMALE` or `MALE` tag to sentences to include gender information. Chu et al. (2017), Kobus et al. (2017), and Stergiadis et al. (2021) used a pseudo-token to specify the target domain for domain adaptation in MT.

In Chapter 6, we use pseudo-tokens to provide timing information to the model to control duration of translated output.

2.2.3 Factored Models

Source Factors

An established method to incorporate linguistic information into phrase-based MT models (Koehn and Hoang, 2007), source factors have also been used in neural MT to add extra information to input sequences. These factors are embedded alongside the source tokens, and all the embeddings are concatenated to form the input embedding to the encoder. Sennrich and Haddow (2016) used factors to provide additional linguistic information alongside the source tokens, such as lemmas, Part of Speech (PoS) tags, morphological features, and dependency labels. Stafanovičs et al. (2020) provided gender annotations as source factors to improve gender translation accuracy. España-Bonet and van Genabith (2018) used factors to model semantic information in the form of synsets to improve translation quality in multilingual MT, especially in zero-shot settings. Dinu et al. (2019) trained models to inject custom terminology by using factors to demarcate inline terminology annotations.

Target Factors

García-Martínez et al. (2016) introduced target factors to decompose the translated output tokens into lemmas and linguistic factors, reducing the target vocabulary size and alleviating the data sparsity problem in generating inflected forms of words. To use factors on the target side, similar to source factors, the embeddings of the tokens and corresponding factors are concatenated before being fed to the decoder. Additionally, there is one output layer per factor, which allows the model to generate each factor separately, and the factors are usually shifted by one position to allow the model to predict the factors conditioned on the corresponding lemma.

Target factors have been used in statistical MT to explicitly model morphology (Bojar, 2007). Apart from decoupling lemmas from linguistic factors (García-Martínez et al., 2016) as described above, they have also been shown to be effective to predict case markers (Nădejde et al., 2017), subword separators (Wilken and Matusov, 2019), capitalisation, or gender information (Niu et al., 2021) decoupled from output tokens. In the area of isochronous MT, they have been used to predict pause markers as an alternative to generating an explicit token (Tam et al., 2022).

While target factors generally do not incorporate any external input information to the model, we introduce the concept of target factors here since in Chapter 6, we present a method to use target factors to provide the model timing information and model the duration of each target token.

2.2.4 Large Language Model Methods

In recent years, decoder-only Large Language Models (LLMs) have rapidly emerged as the predominant paradigm for most NLP tasks, and this extends to MT as well. While this style of model was originally designed for language modelling and text generation (Brown et al., 2020; Radford et al., 2019), LLMs have demonstrated the emergent ability to perform numerous other tasks using prompting and in-context learning methods (Brown et al., 2020; Wei et al., 2022).

Some work (Gao et al., 2023; Jiao et al., 2023) has shown that it is possible to simply provide an instruction, or a prompt, to an LLM to translate an input sentence or document. This method is known as **prompting**, and the prompt can be adjusted to provide any kind of information or guidelines to control the output of the model. For example, Yamada (2023) showed that integrating some information about the purpose of translation and the target audience into prompts can modify and improve translations elicited from ChatGPT. Moslem et al. (2023) provided terminologies as part of the prompt to control the translation of domain-specific terms by LLMs.

Most efforts to perform MT using LLMs (Hendy et al., 2023; Lin et al., 2022; Vilar et al., 2023; Zhang et al., 2023; Zhu et al., 2024) involve providing a few examples to the LLM as part of the prompt, which allows the model to learn the task of translating. This method is known as **few-shot learning**, and the few-shot examples can be chosen or constructed to provide the model with the necessary information to translate as desired. For example, Garcia et al. (2023) showed that carefully selected examples can be used to control regional varieties and formality in translations.

Xu et al. (2024a), Xu et al. (2024c), and Xu et al. (2024b) fine-tune pre-trained LLMs on parallel data to perform MT. While these works typically fine-tune the model with a fixed prompt designed to generically elicit translations, these could also be adjusted to incorporate extra information in the prompt to control the translation.

LLMs' longer context windows and larger model capacity also bring an inherent ability to better capture context when provided. While most of the work discussed above still performs translation at the sentence-level, LLMs have therefore also been used to perform document-level MT in some related work (Karpinska and Iyyer, 2023; Petrick et al., 2023; Wang et al., 2023; Wu et al., 2024; Zhang et al., 2023). Models have been developed to encode extremely long contexts (Bulatov et al., 2024; Chen et al., 2023; Xiong et al., 2024), but are usually not evaluated on MT, since the task arguably does not need context lengths of thousands or millions of tokens.

In addition to the methods described above, there are many other ways to incorporate extra information into MT models, such as memory-augmented MT (Feng et al., 2017) to store and use a translation memory of difficult-to-translate words, retrieval-augmented MT (Bouthors et al., 2024; Bulte and Tezcan, 2019; Cai et al., 2021; He et al., 2021; Hoang et al., 2023; Khandelwal et al., 2021; Xu et al., 2020) to retrieve similar examples to guide translation, etc. Since we do not use these methods in this thesis, we do not go into further detail.

Chapter 3

Cheat Codes to Quantify Missing Source Information

We have established in Chapter 1 that the process of translating a source sentence into a target language accurately requires some supplementary or external information. This chapter is motivated by the question: how can we *quantify the amount of information* that a neural MT model needs in addition to the source sentence? In other words, how much more information does it need to generate a specific reference translation?

For the purposes of this work, we ignore the linguistic significance of the information needed by the model, instead approaching the problem in a purely abstract way. Our goal is not to interpret the content of the required information in any way, but only to quantify it numerically.

To this end, we describe a method to empirically estimate the amount of information $H(t|s)$ added by the target sentence t that is not present in the source s in a neural MT system. We do this by providing the model the target sentence in a highly compressed form (a "cheat code"), which due to its limited size, should encode target-side information that the model cannot already obtain from the source encoding.

Exploring the effect of the size of the cheat codes, we find that the model is able to capture some useful extra information from just a single float representation of the target and nearly reproduces the target with two 32-bit floats per target token. This shows us that a suprisingly small amount of extra information needs to be provided to guide a model to a perfect translation.

This method does not intend to present us with a way to actually provide any of this missing information to the MT model since the target sentence is not available at inference time in a real translation scenario. It is instead an analysis method that helps us understand the informational content of the target sentence that is absent from the source sentence.

The content of this chapter is based on work published at NAACL 2022 (Pal and Heafield, 2022).

3.1 Motivation

Given a sentence s in the source language, a machine translation system generates a translation t in the target language. However, as discussed in Chapter 1, for any sentence of non-trivial complexity being translated between most pairs of natural languages, the translation t is not unique. Therefore, to reproduce a reference translation, a model requires some amount of extra information.

While earlier neural models were constrained in terms of model capacity and computational tractability, this is no longer a significant problem with modern hardware and models for MT. It is therefore unlikely that current neural MT models' inability to generate specific translations is due to a lack of representational capacity. It is of course clear that not all the necessary information is present in the sentence-level input that is being provided to these models, but it is not clear *how much* information is missing. By quantifying this information, we can gain a better understanding of the limitations of current models, and have a better idea of whether this information can realistically be provided to the model and whether the models can capture it efficiently.

The aim of this chapter is therefore to establish a method to quantify the amount of information that is missing in the source s that is required to generate the translation t. This is analogous, but not formally equivalent to $H(t|s)$ in information theoretic terms. Our analysis does not account for valid paraphrases or synonyms. Therefore, what we measure is not just how much information is needed to generate *any correct translation*, but how much information is needed to generate a *specific* translation. Factors like style and lexical choice that do not directly affect correctness are also an integral part of the translation, and we want to measure the information required to capture these aspects as well.

To quantify this information, we modify the standard encoder-decoder model architecture (Section 2.1) to provide the target sentence to the model as an auxiliary input, and observe the effect of varying the size of the representation of the target sentence. We vary the size ranging from the minimum that provides any useful information at all to the decoder, all the way up to the size that enables a near-perfect reproduction of the target. At the lower end, this informs us about what the smallest amount of extra information is that is still useful to the model. At the other end of the range, this allows us to estimate an upper bound on the amount of additional information present in a specific translation that it cannot capture from the source.

Since the target being exposed to the model is generally considered a form of "cheating" in machine learning, we refer to these compressed representations of the target as "cheat codes".

3.2 Related Work

Voita et al. (2021) used a variant of Layerwise Relevance Propagation (LRP) to measure the relative contribution of the source sentence and the target prefix while generating a specific target token. This does not reflect on information from outside of the current sequence being translated. Bugliarello et al. (2020) have previously proposed an information-theoretic measure of the difficulty of translating sentences from one language to another called cross-mutual information (XMI). Inspired by this metric, Fernandes et al. (2021) proposed conditional cross-mutual information (CXMI) to measure how much document context is used by a document-level MT model. Mohammed and Niculae (2024) also measured context usage by document-level MT models, but by perturbing the input context and observing the effect on the output, instead of from an information-theoretic perspective.

To the best of our knowledge, ours is the first work that attempts to quantify the amount of additional information required by neural MT models – not just from context, but from any possible source that is not the input sentence.

3.3 Method

3.3.1 Model Architecture

We use a modified dual-encoder Transformer architecture (Zoph and Knight (2016); see Section 2.1) similar to the one used by Junczys-Dowmunt and Grundkiewicz (2018), but without the tied encoder parameters.

The first encoder is a standard Transformer-base encoder (Vaswani et al., 2017) which takes the source sentence as input, while the second encoder generates a highly compressed representation of the second input. The decoder attends to both encoder contexts – each decoder layer has a multi-head attention block for each encoder and these blocks are stacked (see Section 2.2.1 and Figure 1 in Junczys-Dowmunt and Grundkiewicz (2018)). Figure 3.1 shows our model architecture along with the separate inputs and cheat codes.

For the second encoder, we use a GRU[1] (Cho et al., 2014b) with hidden size 256, optionally average its outputs over all the states to get a fixed-length representation, and apply a linear bottleneck layer. This generates the highly compressed representation of the second input, or the cheat code, that the decoder attends to.

The model was implemented using the Marian framework (Junczys-Dowmunt et al., 2018), and all the code including training scripts are publicly available[2].

This architecture does not allow the second encoder access to the source; in other words, the cheat code is encoded with no knowledge of the corresponding source sentence. This forces the second encoder, given its limited representation capacity, to guess what information from the target is most likely to not be present in the source and encode it. An alternative architecture could provide access to both source and target sentences in the second encoder, which would allow the second encoder to analyse the source sentence to decide what information is actually missing. However, in this alternative architecture, it would be impossible to separate the source information from the target information, and there would be no way of preventing the model from using the extra capacity of the second encoder to encode aspects of the source as well, making the cheat codes much less informative. So we choose to use two completely separate encoders in this work, with the assumption that the models learn to encode the target sentence in an efficient way that allows it to extract the maximum amount of information from the cheat code that would not already be available in the source.

[1] We note that the choice of GRU as the architecture for the second encoder is not significant. It could also be a Transformer encoder.

[2] https://github.com/Proyag/marian-dev/tree/cheat-codes

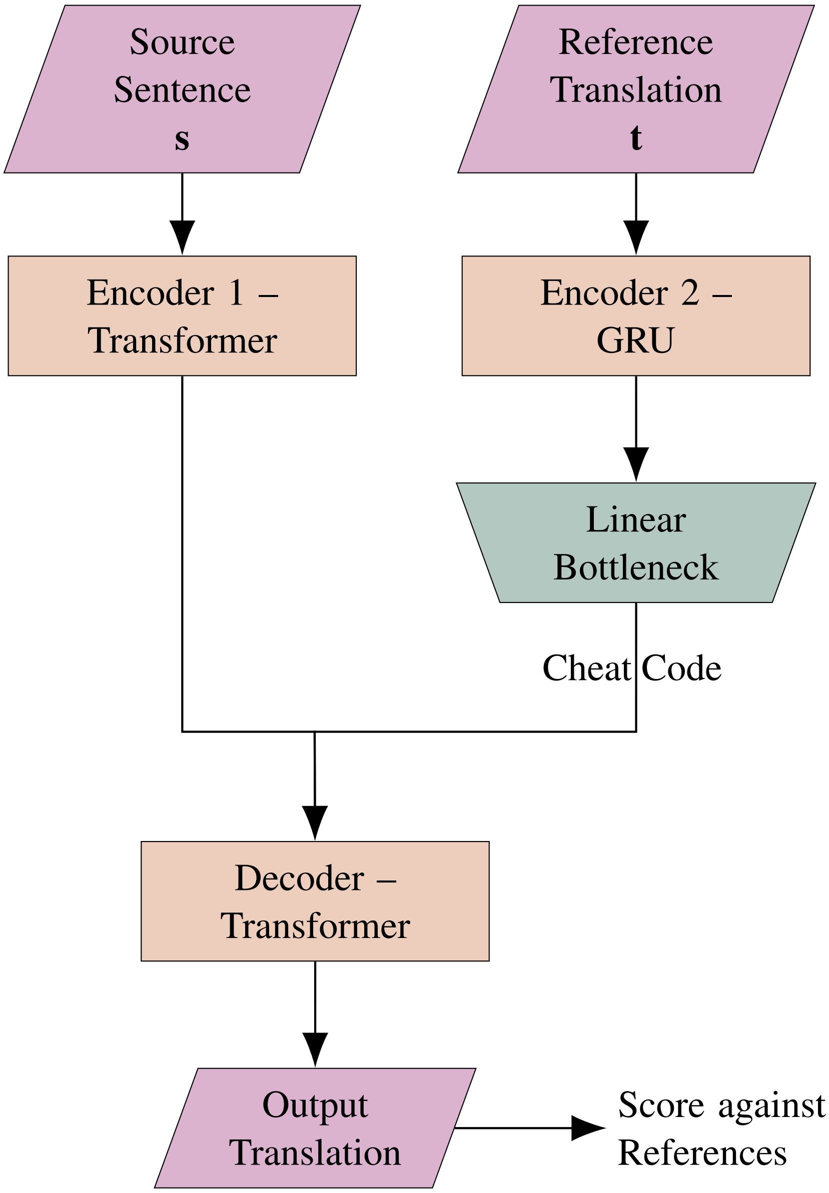

Figure 3.1: Model architecture showing inputs and cheat codes.

3.3.2 Cheat Codes

At training time, we provide the target sentence as the second input to the model, so the model essentially cheats by seeing the translation it is supposed to generate. At inference time, we can provide the reference translation or any other sentence as the second input, which should guide the generation towards this provided input.

Alternatively, this second encoder can be bypassed to directly provide context vectors for the decoder to attend to. As an example, we can use this feature to interpolate between the representation of two different references and provide that as a cheat code, and thus explore whether we can obtain alternative translations in some semantic space between the two references (Section 3.4.4).

We vary the size of the cheat codes and observe their effect on the output translations. The size is varied in three different ways:

Fixed-length cheat codes Using fixed-length representations of n dimensions (essentially floating-point numbers), where we can vary n. Given a target sequence of length $|Y|$ with d-dimensional output states from the second encoder, we average over all the $|Y|$ output states of the second encoder, and then apply the bottleneck layer to project the result down from d to n dimensions.

Variable-length cheat codes Using variable-length representations of n floating-point numbers per token, which is simply the output of the second encoder with no averaging over time steps, and the same bottleneck layer applied separately on each output state to project d dimensions to n, resulting in cheat codes of size $n \times |Y|$.

Quantisation Using representations smaller than one floating-point number by applying quantisation on a one-dimensional representation. We do this using a simple linear quantisation scheme similar to Miyashita et al. (2016) and Hubara et al. (2017). To quantise a scalar x to k bits:

$$\begin{aligned} r &= \text{round}(x * m) \\ c &= \text{clip}(r, -2^{k-1}, 2^{k-1} - 1) \\ \text{Quant}_k(x) &= c/m \end{aligned}$$

where m is a multiplier chosen to ensure the quantised scalar covers the full range of the k-bit number after quantisation. We observe that our single float32 cheat codes are in [-2, 2], so we use $m = 2^{k-2}$ so that r is spread over the full range of $[-2^{k-1}, 2^{k-1}]$ without getting clipped.

One possible concern is that the cheat codes might learn to encode the entire target sequence and the model could just copy the target and ignore the source entirely. However, it has been observed that even with much larger sentence representations such as 1024-dimensional LASER (Artetxe and Schwenk, 2019) and 768-dimensional LaBSE (Feng et al., 2022), it is non-trivial to auto-encode the target sentence (Duquenne et al., 2022). We therefore expect that our small target representations will encourage the model to fully use the source information and use the limited additional capacity of the cheat code to encode the additional information that it requires. We verified this by testing the models with empty source inputs and a valid cheat code, and observed that the models are not able to generate the translations just from the cheat code. In the case of variable-length cheat codes, the capacity might be large enough to begin to auto-encode the target, and so we follow a two-step training process designed to avoid the copying behaviour (details in Section 3.4.3).

3.4 Experiments

We train strong models on a high-resource language pair – German→English, and measure the effects of varying the size of the cheat codes, i.e. the amount of additional information available to the model, on the quality of the generated translations. We use Chen et al. (2021)'s cleaned version of the WMT21 German→English dataset (Akhbardeh et al., 2021) for all experiments. We do not use back-translated data since we observed no improvement in quality upon adding it, consistent with Chen et al. (2021)'s findings.

The WMT21 German→English test sets included multiple references produced by different translators. We evaluate on two of these references, labelled A and B, since we represent these references as cheat codes and evaluate their effect on the output translations, and we do not want our findings to be specific to references from a particular translator.

As evaluation metrics, we use BLEU[3] and ChrF[4] metrics from SacreBLEU (Post, 2018), and COMET and COMET-QE[5] (Rei et al., 2020).

Table 3.1 shows the results for our different models with references A or B provided as cheat codes and being evaluated on both references. We see that the models can score higher than the Transformer baseline on a given reference when the same reference is supplied as a second input, which indicates that the model is able to "cheat" and capture useful extra information from just a single floating-point representation of the target sentence.

3.4.1 Increasing bottleneck size

We gradually increase the size of the cheat codes to observe how the translation quality improves with more additional information from the target, and how much information the model needs to approximately reproduce the reference translations.

As we increase the size of the bottleneck layer, we see (Table 3.1; Figure 3.2) that the model captures more information from the larger cheat codes and the outputs approach the reference translations, as shown by much higher BLEU and ChrF compared to the baseline. However, this is not always reflected in the COMET and COMET-QE scores and we suspect this is due to how COMET is trained. This issue is further discussed in Section 3.4.5.

We also note that even with large variable-length cheat codes, the model struggles to achieve perfect scores, showing that they are unable to generate the exact references even with a large amount of additional information. This is further studied in Chapter 4.

3.4.2 Minimising bottleneck size

We also explore the lower bound of the size of cheat codes that the models can meaningfully capture information from. By showing that even very small cheat codes are informative, we can establish that the information required by the model can be encoded very efficiently, and that encoding dimensions or model capacity is unlikely to be a constraint for MT models to capture this information.

[3] `BLEU|#:1|c:mixed|e:no|tok:13a|s:exp|v:2.0.0`

[4] `chrF2|#:1|c:mixed|e:yes|nc:6|nw:0|s:no|v:2.0.0`

[5] `wmt20-comet-da` and `wmt20-comet-qe-da` in COMET

Model/Input	Score on Reference A			Score on Reference B			
	BLEU	ChrF	COMET	BLEU	ChrF	COMET	QE
Transformer baseline	32.2	60.3	0.5565	36.3	62.6	0.5640	0.3472
References scored against each other / with COMET-QE:							
Reference A	100	100	0.9934	29.5	58.5	0.5316	0.3265
Reference B	29.5	57.7	0.5643	100	100	1.0015	0.3829
Reference A as second input, fixed-length cheat codes:							
1 × int4	31.1	58.9	0.4781	31.8	59.0	0.4610	0.2924
1 × int8	31.3	59.1	0.4885	31.0	58.8	0.4707	0.3067
1 × int16	32.0	59.7	0.5320	31.2	59.2	0.4913	0.3107
1 × float32	32.3	59.6	0.5153	31.6	59.2	0.4917	0.3092
2 × float32	33.5	60.3	0.5177	29.6	58.2	0.4602	0.2979
4 × float32	36.7	61.6	0.4935	27.0	56.3	0.3893	0.2558
8 × float32	40.7	63.7	0.5023	25.1	54.9	0.3206	0.2235
12 × float32	47.0	67.4	0.5202	23.7	53.9	0.2790	0.2245
16 × float32	57.2	73.3	0.6553	24.4	54.0	0.3100	0.2404
25 × float32	67.0	80.0	0.7333	24.6	54.4	0.3191	0.2561
Reference A as second input, variable-length cheat codes:							
1 × float32 / token	40.1	64.2	0.5962	28.7	57.8	0.4587	0.2948
2 × float32 / token	92.4	96.1	0.9148	28.4	57.6	0.4473	0.2778
4 × float32 / token	91.2	95.2	0.9017	28.5	57.6	0.4434	0.2773
8 × float32 / token	89.7	94.1	0.8877	28.6	57.6	0.4438	0.2810
12 × float32 / token	94.1	97.4	0.9377	28.6	57.8	0.4750	0.2971
16 × float32 / token	95.8	98.6	0.9779	28.7	57.9	0.5107	0.3152
25 × float32 / token	93.9	96.8	0.9211	28.6	57.5	0.4526	0.2888
32 × float32 / token	96.6	98.7	0.9593	28.7	57.9	0.4720	0.2920
Reference B as second input, fixed-length cheat codes:							
1 × int4	29.8	58.0	0.4624	34.5	60.5	0.4735	0.2981
1 × int8	28.9	57.9	0.4824	34.9	60.6	0.5147	0.3121
1 × int16	29.1	57.9	0.4942	36.3	61.7	0.5375	0.3145
1 × float32	29.3	58.2	0.4865	36.4	61.9	0.5153	0.3111
2 × float32	27.5	57.0	0.4706	38.3	62.9	0.5249	0.3056
4 × float32	25.7	55.6	0.4210	41.8	64.4	0.5344	0.2827
8 × float32	24.6	54.3	0.3677	46.6	67.1	0.5500	0.2621
12 × float32	24.1	53.8	0.3354	54.3	71.5	0.6147	0.2562
16 × float32	24.3	53.6	0.3510	62.8	76.3	0.6995	0.2771
25 × float32	24.9	53.9	0.3657	70.7	81.8	0.7734	0.2899
Reference B as second input, variable-length cheat codes:							
1 × float32 / token	26.9	56.6	0.4725	46.0	67.0	0.6275	0.3125
2 × float32 / token	28.4	56.7	0.4785	92.5	95.5	0.9130	0.3234
4 × float32 / token	28.7	57.0	0.4959	92.0	95.3	0.9156	0.3303
8 × float32 / token	28.6	56.8	0.4919	90.6	94.4	0.8997	0.3320
12 × float32 / token	28.7	57.0	0.5123	94.0	96.9	0.9514	0.3439
16 × float32 / token	28.7	57.0	0.5349	95.6	98.0	0.9783	0.3599
25 × float32 / token	28.8	57.0	0.5082	93.8	96.4	0.9331	0.3438
32 × float32 / token	28.7	57.0	0.5097	96.1	98.0	0.9576	0.3468

Table 3.1: Evaluation with references A and B as second input.

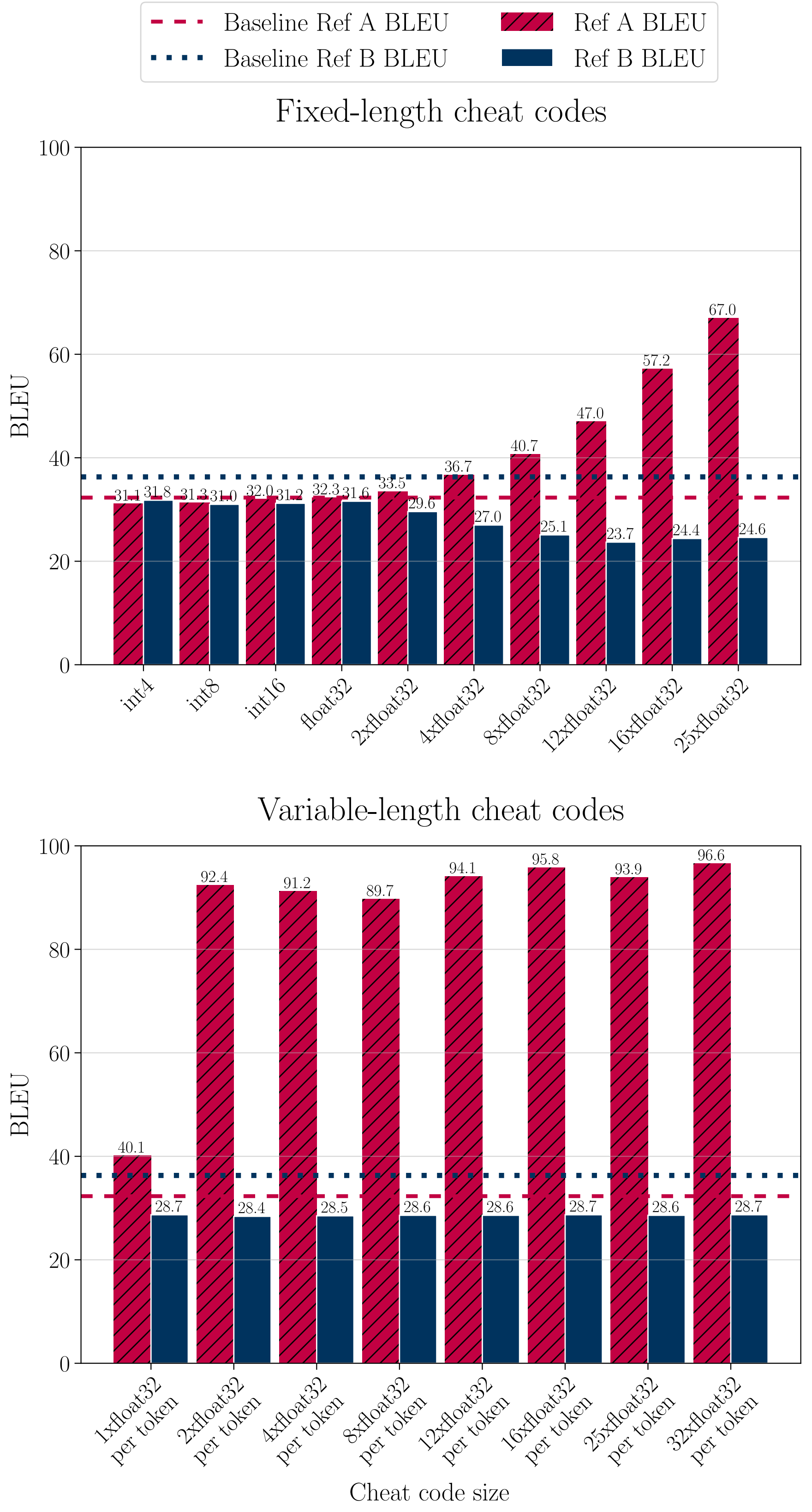

Figure 3.2(a): Results with reference A as the second input.

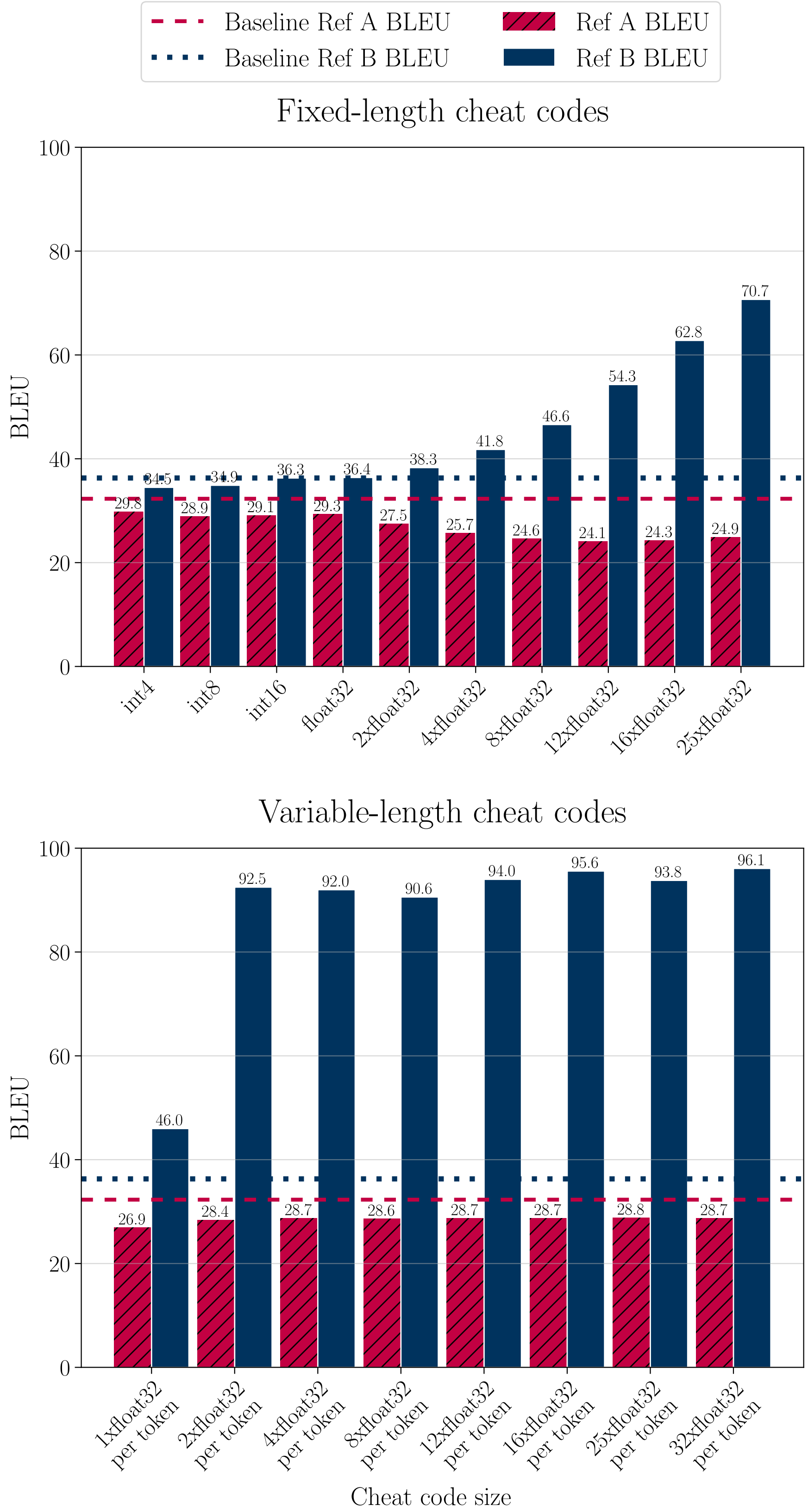

Figure 3.2(b): Results with reference B as the second input.

We observe that the model may be able to capture useful information from a single 32-bit float (Table 3.1, rows with 1×float32), as shown by a very slight increase of the BLEU score over the baseline. The effect is much more visible from cheat codes of size 2×float32. To estimate the lower bound of the cheat code size that might still be informative to the model, we reduce them even further. To reduce the size below a single float, we quantise the 32-bit representations from the second encoder to 16, 8, or 4 bits. We see that the 16-bit cheat codes (Table 3.1, rows with 1×int16) work almost as well as the 32-bit ones. With less than 16 bits, it appears that the model is unable to capture any measurable information from the target.

We note here that there is no guarantee or even expectation of optimality of representation in the cheat codes, i.e. there is always the possibility that there exists an alternative way to encode the required information in a more efficient way. Therefore, while we observe that cheat codes smaller than a certain size do not appear to convey enough meaningful information to measurably improve translation accuracy, an optimal cheat code could well be smaller and still benefit translation. What we are able to measure is empirically the smallest effective cheat codes for our models, but we do not make any claims about how small an optimal cheat code could be.

3.4.3 Variable-length cheat codes

Since the amount of information contained in sentences can vary widely, it makes sense that the size of cheat codes required to encode them can vary. To this end, we also train models where the size of the cheat code is proportional to sentence length, as described in Section 3.3.2.

For these models, we observe that due to the increased capacity of the second encoder, training a model to "cheat" from the start makes it too dependent on the target, i.e. it begins to learn simply to copy the entire target instead of using the source fully, effectively resulting in the cheat code estimating $H(t)$ instead of $H(t|s)$ as intended. Therefore, we adopt a two-step training process: first, we train the model with a blank second input (i.e. no leaked target information) for the model to learn to fully use the source, then we continue training with both inputs to train the second encoder to capture the additional information. We also validate this process at test time by providing a blank source input and confirming that the model is unable to copy any meaningful translations solely from the second encoder.

As expected, we observe (Table 3.1; Figure 3.2) a similar pattern of more information being captured as we make the cheat codes larger. At just 2 floats per token (rows with 2×float32/token), the model scores 92.4 BLEU/96.1 ChrF on reference A with the same reference as input, and likewise 92.5 BLEU/95.5 ChrF on reference B. At 16 floats per token, it scores more than 98 ChrF, which is very close to perfectly reproducing the references.

3.4.4 Interpolating between references

We speculate that it could be possible to control the output of these models in other ways than just to reproduce the references by manipulating the cheat codes. To test this hypothesis, we can directly feed the decoder a modified cheat code instead of the representation of an actual target sentence. For models which use fixed-length representations of the second input, we can interpolate between the encoded forms of the two references. This is not possible for variable-length cheat codes since different references do not necessarily have the same length. Given a small enough fixed-length representation, we can even sweep through the entire range of cheat codes and possibly produce a large number of different but potentially valid translations. We hypothesised that we might be able to generate diverse high-quality translations (He et al., 2018; Roberts et al., 2020).

Figure 3.3 shows the performance of the model with fixed-length cheat codes of size 2×float32, while providing $\lambda \cdot \text{enc(refA)} + (1-\lambda) \cdot \text{enc(refB)}$ as the cheat code. We can see the emergence of a continuous space of cheat codes such that codes close to reference A result in outputs similar to reference A and gradually moving the code towards reference B makes the output increasingly similar to reference B.

Unfortunately, preliminary experiments showed that using the interpolated cheat codes to generate modified outputs produces fragmented or even nonsensical translations, instead of any meaningful interpolations between varied references.

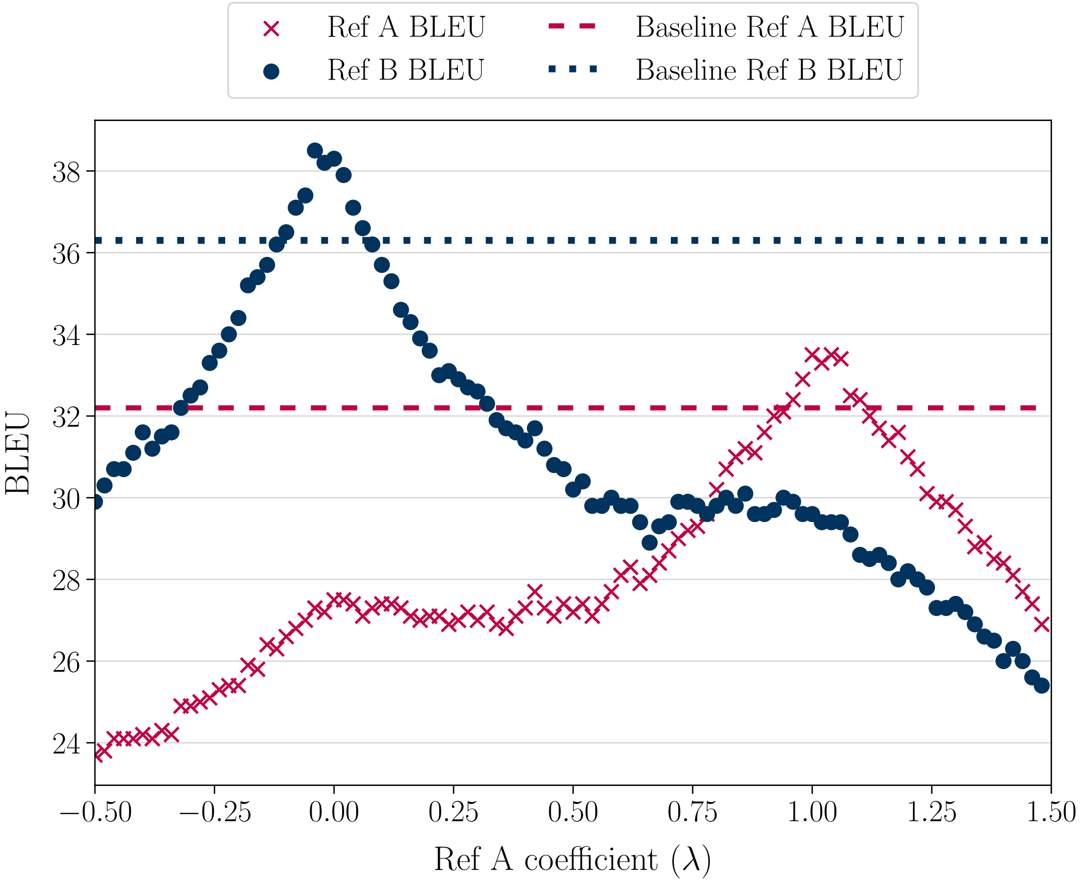

Figure 3.3: Test set results when providing cheat codes that are interpolated between representations of references A and B using fixed-length cheat codes of 2 floats. The plots show how the model scores highest on reference A when the cheat code is close to the representation of reference A, and similarly for reference B.

3.4.5 Evaluating with COMET-QE

BLEU and ChrF, along with most commonly used MT metrics, are reference-based metrics. This automatically makes it more likely that the model will score highest on a reference when given that exact reference as the cheat code. In Figure 3.3, for example, we see how the performance on each reference peaks exactly when we provide that reference as input. Since the two references are quite different from each other – they only score 29.5 BLEU when they are scored against each other – using one as the cheat code does not produce good results on the other.

We expected to see COMET-QE scores increase with cheat code size, similar to BLEU and ChrF scores. However, we see that COMET-QE scores remain below the baseline even for most models with large cheat codes, which have near-perfect BLEU/ChrF and high COMET scores. We even observe that COMET-QE scores Reference A lower than the baseline output. Previous meta-evaluation work has already shown that reference-free metrics show poor or even negative correlation with translation quality in certain situations. We conclude that since COMET-QE is a metric trained on usually flawed MT outputs and their human evaluation scores, it does not work well for near-perfect translations and is unable to score them higher than the best previously observed MT output. For the same reason, even though COMET scores (with reference) increase for large cheat codes, the pattern is less clear than for the string-matching metrics.

3.5 Relation to Cross-entropy

While we do not approach our analysis from an information theoretic point of view, in this section, we briefly discuss the relationship of our method to the concepts of cross-entropy and conditional entropy.

Cross-entropy is commonly used in NLP tasks as a measure of the discrepancy between the predicted probability distribution of a model and the true distribution. Specifically for MT, this is the difference between the translation probability distributions generated by the model and the reference translations. A lower (better) cross-entropy indicates that the model predictions are closer to the reference translations.

Conditional information content is a measure that describes the uncertainty in a target t given the source s. Conditional entropy, which is the average conditional information content over all s and t, is therefore an indicator of the average amount of information missing from the source that is required to generate the specific targets. Lower conditional information content indicates that the translation of a source sentence is less ambiguous and requires less additional information to generate the target.

Our method does not allow the quantification of different amounts of conditional information content for individual sentence pairs, but rather determines a cheat code size that is applied across all sentence pairs to measure the overall translation quality over the entire test set. The cheat code size that enables the model to reproduce target sentences can therefore be interpreted as an upper bound on the conditional information content needed to generate the reference translations.

We note that our work does not strictly measure information content in an information theoretic sense, since it is confounded by other factors such as the model's ability to translate accurately irrespective of missing source information, encoding inefficiencies of the model encoders, and the model potentially learning to partially copy the target from the cheat code. However, the method still enables us to establish an approximate upper bound on the amount of missing source information.

3.6 Conclusions and Future Work

This chapter has shown that by letting machine translation models use a highly compressed representation of the target sentence as an auxiliary input, we can estimate the amount of information missing from the source that the model needs and can capture about the target translation. By varying the size of these representations (or cheat codes), we see that the model can capture useful information from representations of the target that are as small as a 16- or 32-bit scalar. We also see that the model approaches perfect reproduction of the target (scoring >92BLEU/95ChrF over a test set) from as little as 2 floats per target token.

However, we find that as the cheat code size increases beyond that, the model quality plateaus and never manages to reach the point where the reference translations are exactly reproduced. This indicates that there are some aspects of the target that the model finds difficult to encode and generate, even with a large amount of guidance in the form of a cheat code. We explore these phenomena in Chapter 4, giving us insight into some aspects of translation that models find hard to learn.

A limitation of our method is that it can only estimate the amount of missing information from the source based on the size of cheat code, but we do not get any qualitative insight into what this information actually is, or how it enriches the translation process. In future work, this method can potentially be extended to qualitatively analyse what the missing information is. One possible direction of future work is to condition the size of the cheat codes on the source sentence, which would force the model to assess how much information it actually needs to translate the source sentence, providing an indicator of the difficulty of an individual input.

The cheat code method provides us an effective analysis tool to measure the amount of additional information, not only for MT but potentially for any sequence-to-sequence task where the output is not uniquely determined by the input, such as summarisation or question answering.

However, the method does not lead us to a way to provide this information to an MT in a realistic scenario where the target is not already available. In Chapter 5 and Chapter 6, we look at ways to provide specific types of information to the model that directly benefit the translations.

Chapter 4

Cheating to Identify Hard Problems for Neural MT

In Chapter 3, we introduced “cheat codes” as a method of leaking varying amounts of target information to a model, thus allowing the model to capture this information and attempt to reproduce a target translation. However, we found – somewhat surprisingly – that despite using cheat codes of large dimensions, the models are usually unable to perfectly reproduce all translations. As seen in Table 3.1, the scores on the test set do not reach perfect automatic metric scores even for the models with the largest cheat codes in our experiments. This leads us to believe that, even with significant amounts of additional information provided to these MT models, some phenomena are inherently difficult for them to learn, and we can use cheating as a tool to identify and analyse these phenomena.

In this chapter, we identify hard problems for neural machine translation models by analysing progressively higher-scoring translations generated by letting models cheat to various degrees. If a system cheats by looking at part of the output and still gets something wrong, that suggests it is a hard problem. Using this method, we are able to identify fine-grained categories of words and phrases that remain challenging for neural MT models, laying the groundwork for future research to address these specific challenges to further improve translation quality.

We use multiple models at varying degrees of cheating (Section 4.4), ranging from a baseline Transformer model to those almost able to reproduce the target, to output translations. We then analyse the accuracy of these outputs in relation to word frequencies (Section 4.5.1), parts of speech (Section 4.5.2), and named entities (Section 4.5.3), to find the types of words and sentences that are more difficult for these models to learn to translate accurately.

Contrary to popular belief, we find that the most frequent tokens are not necessarily the most accurately translated due to these often being function words and punctuation that can be used more flexibly in translation, or content words which can easily be paraphrased. We systematically analyse system outputs to identify categories of tokens which are particularly hard for the model to translate, and find that this includes certain types of named entities, subordinating conjunctions, and unknown and foreign words (Section 4.5).

We also encounter a phenomenon where words, often names, which were not infrequent in the training data are still repeatedly mistranslated by the models — we dub this the “Fleetwood Mac problem” (Section 4.5.4).

The content of this chapter is based on work published in the Findings of EACL 2023 (Pal and Heafield, 2023).

4.1 Motivation

While neural MT generally works well given enough high-quality data to train on, the prevalence of automatic evaluation metrics and human evaluation frameworks which produce a single number as a measure of quality causes research to gloss over more fine-grained problems. While the field is generally aware of some high-level hard problems for neural MT (Koehn and Knowles, 2017; Wan et al., 2022), our goal here is to analyse MT output to identify the types of words and phrases that are more difficult for neural MT to translate accurately.

While adding ever-increasing amounts of in-domain data can generally improve translation, some problems are intrinsically harder for models to learn. This chapter aims to identify some of these hard problems for MT that are likely to remain challenging even with larger in-domain datasets.

The way we approach this is to allow the models to cheat. In Pal and Heafield (2022) (see Chapter 3), we introduced a method to provide a highly compressed representation of the desired translation (a “cheat code”) as an auxiliary input to the model so that the generated output is encouraged to be closer to the target output. While that work was motivated as a method to estimate the amount of information present in the target that is missing in the source, we adopt the same method to produce unrealistically accurate models, and contend that if the models get particular things wrong even with hints from cheat codes, those are the relatively difficult things to translate.

We also use a second method of cheating — fine-tuning a standard MT model on the test set — with the motivation that if we observe models with different methods of cheating showing similar errors in translation, it is reasonable to conclude that those errors are genuinely difficult things to translate and not just quirks of how the cheating method affects the translation. While large amounts of in-domain data can improve overall quality significantly (Edunov et al., 2018), this fine-tuning method lets us expose the model to the most relevant data possible: the test set itself. The longer we fine-tune, the more it learns to cheat and becomes more accurate on the test set. Translations that cannot be learned correctly from the test set itself are very unlikely to be learned from adding arbitrarily large amounts of in-domain data, and can be identified as difficult problems to be targeted in future research.

Using these two methods of cheating (described in more detail in Section 4.3), we can vary how much the models cheat and observe what parts of sentences and types of words are easier to translate with increasingly accurate models. This tells us about which parts of translation take the most cheating to learn, and thus signpost some hard problems for neural machine translation.

4.2 Related Work

Automatic MT evaluation metrics such as BLEU (Papineni et al., 2002), chrF (Popović, 2015), METEOR (Banerjee and Lavie, 2005), COMET (Rei et al., 2020), and BLEURT (Sellam et al., 2020) exist in abundance, but a more fine-grained view of the errors made by translation systems is often required to determine weaknesses of models. Vilar et al. (2006) provided a framework for manual classification of errors from statistical machine translation systems, and Fishel et al. (2011), Zeman et al. (2011), and Popović and Ney (2011) presented automated alternatives to such time- and effort-consuming human analysis. Burchardt (2013) introduced the Multidimensional Quality Metrics (MQM) framework for systematic error analysis of translation outputs.

Koehn and Knowles (2017) presented a high-level analysis of challenges for neural MT. There are also methods to evaluate specific aspects of machine translation, such as contrastive translations to evaluate pronoun translation (Müller et al., 2018), transliteration or morphosyntactic agreement (Sennrich, 2017), and challenge sets (Isabelle et al., 2017; King and Falkedal, 1990). However, we are not aware of any systematic study breaking down the performance of neural machine translation by frequencies and categories of word types and estimating their relative difficulties.

Phenomena such as rare words and named entities being inaccurately translated are considered common knowledge and numerous works (Jean et al., 2015; Koehn and Knowles, 2017; Luong et al., 2015b; Sennrich et al., 2016c) have offered various solutions to the problem. Subword segmentation (Kudo, 2018; Sennrich et al., 2016c) is the most commonly used method to improve the translation of rare words, but Sennrich et al. (2016c)'s analysis also showed that while it significantly improves the translation of conjugated and compound words, the models still struggle with names due to inconsistent segmentation and ambiguous transliteration. Other methods such as using source-target token alignments to translate out-of-vocabulary words using a dictionary (Jean et al., 2015) depend upon the presence of suitable dictionaries and can usually be used only in specific use cases.

Tools such as compare-mt (Neubig et al., 2019) and MT-Telescope (Rei et al., 2021) aggregate different kinds of analyses based on token frequencies, types of words, and linguistic labels (such as parts of speech or named entities) together into reports to provide a detailed view of the errors in MT output, which we use for our purposes.

4.3 Cheating Methods

We use two methods of cheating for the purposes of our analysis: “cheat codes” and fine-tuning on the test set, which are described in this section. The idea is to use two different methods of cheating as a way to separate the analysis of which problems are actually difficult for neural MT from that of the cheating methods themselves.

4.3.1 Cheat Codes

The first method of cheating is the use of "cheat codes" (Pal and Heafield (2022); Chapter 3), which are bottlenecked representations of the target sentence provided as an additional input to the model. As shown in Figure 3.1 (page 27), a dual-encoder architecture (Junczys-Dowmunt and Grundkiewicz, 2018) is used, i.e. the Transformer architecture (Vaswani et al., 2017) is augmented with a second GRU encoder (Cho et al., 2014b), which takes the target sentence as its input, followed by a linear layer which bottlenecks the generated target representation to a much smaller size, of the order of a few floats. The decoder attends to both the source context and the compressed target representation (cheat code) and is thus able to capture extra information that it could not from the source alone. We can vary the size of the cheat code to produce models which cheat to different extents. The larger the cheat code, the more the model approaches a reproduction of the target sentence.

4.3.2 Fine-tuning on the Test Set

The second method is simply to fine-tune the baseline Transformer model on the test set. We validate and save checkpoints every 10 updates where each update is performed on a single batch consisting of the entire 1000-line test set. We use the outputs obtained from these checkpoints to analyse the gradual change in performance.

4.4 Models

4.4.1 Baseline

Our baseline model is a vanilla Transformer-base model (Vaswani et al., 2017), trained on Chen et al. (2021)'s cleaned version of the WMT21 German→English dataset (Akhbardeh et al., 2021). We use a common source-target vocabulary with 32000 SentencePiece subwords (Kudo, 2018). As observed by Chen et al. (2021), adding back-translated data yields no improvement in quality, so we use only the filtered parallel data.

We evaluate on reference A of the WMT21 test set using BLEU[1] and ChrF[2] metrics from SacreBLEU (Post, 2018), and COMET[3] (Rei et al., 2020). Since our analyses are based on exact word matches and paraphrases or synonyms are not considered matching translations, we focus mainly on the string-matching metrics, especially BLEU. However, in general we find that the trends observed are consistent across all metrics.

4.4.2 Models using Cheat Codes

We use models with cheat codes of varying sizes from Chapter 3 – larger representations of the target as the auxiliary input mean the model produces translations closer to the desired target. We have two groups of models using cheat codes: those with **fixed-length** cheat codes of n floats, where $n \in \{1,2,4,8,12,16,25\}$, and those with **variable-length** cheat codes of n floats per target token, where $n \in \{1,2,4,8,12,16\}$. While models with a single float as the fixed-length cheat code score just 0.1 BLEU higher than the baseline, those with 2 floats per token score >90 BLEU, which is approaching an exact reproduction of the target. Table 4.1 shows all the models with different cheat code sizes along with their overall quality.

4.4.3 Models Fine-tuned on the Test Set

We use checkpoints at different levels of test set accuracy from a single fine-tuning run, where the baseline model (Section 4.4.1) is fine-tuned on the test set, with reference A on the target side. We have 94 such checkpoints, one for every 10 updates. Figure 4.1 shows the evolution of the scores on the test set over the fine-tuned checkpoints. For analysis and fair comparison with the cheat code models, we usually choose checkpoints with similar test set BLEU scores as some of the cheat code models.

[1] `BLEU|#:1|c:mixed|e:no|tok:13a|s:exp|v:2.0.0`

[2] `chrF2|#:1|c:mixed|e:yes|nc:6|nw:0|s:no|v:2.0.0`

[3] wmt20-comet-da in COMET

Model/input	BLEU	ChrF	COMET
Baseline	32.2	60.3	0.5565
Fixed-length cheat codes:			
1 float	32.3	59.6	0.5153
2 floats	33.5	60.3	0.5177
4 floats	36.7	61.6	0.4935
8 floats	40.7	63.7	0.5023
12 floats	47.0	67.4	0.5202
16 floats	57.2	73.3	0.6553
25 floats	67.0	80.0	0.7333
Variable-length cheat codes:			
1 float / token	40.1	64.2	0.5962
2 floats / token	92.4	96.1	0.9148
4 floats / token	91.2	95.2	0.9017
8 floats / token	89.7	94.1	0.8877
12 floats / token	94.1	97.4	0.9377
16 floats / token	95.8	98.6	0.9779

Table 4.1: Test set scores for all the cheat code models used for analysis.

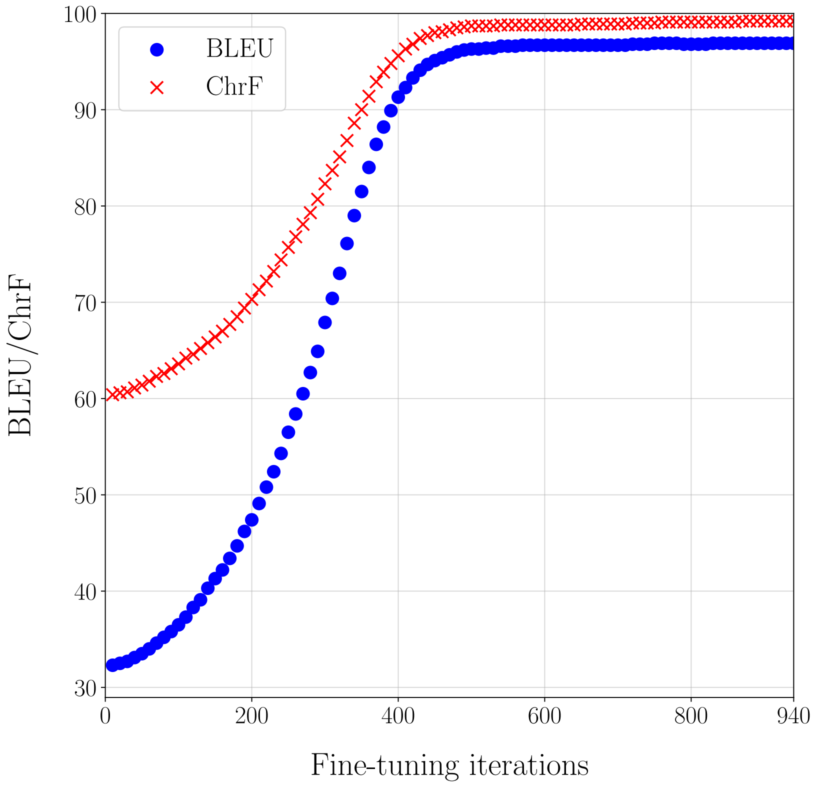

Figure 4.1: Evolution of test set scores with fine-tuning on the test set.

4.5 Analysis

We analyse the outputs of our models to identify the types of errors that they make, and how the frequency of these errors change with increasing model quality induced by cheating. We focus on the accuracy of translation by token frequency, parts of speech, and different categories of named entities.

We use `compare-mt`[4] (Neubig et al., 2019) to systematically analyse and compare the outputs of the different models. We use the `normalize-punctuation.perl`[5] script from Moses (Koehn et al., 2007) to normalize punctuation on the target side before analysis. For PoS tagging and Named Entity Recognition (NER) in English, we use the RoBERTa-based (Liu et al., 2019) `en_core_web_trf`[6] model from `spaCy`. Since the same trends are usually observed irrespective of the method of cheating, we present most findings for one method, and a comparison of the methods in Section 4.5.5. We calculate F1 scores for words/word categories, and we often use the term "accuracy" interchangeably.

We point out that these analyses are done at the word level using exact string matching, so they do not account for the fact that a word can be translated correctly even if it is not an exact match. However, through our manual inspection of the outputs, we find that many of the errors are real mistakes and not paraphrases/synonyms, so the results of our analyses are still meaningful and informative.

4.5.1 Token Accuracy by Frequency

The translation of rare words is a well-known problem in machine translation (Koehn and Knowles, 2017; Luong et al., 2015b; Vanmassenhove et al., 2019; Zhang et al., 2022), which a large body of research work has partially addressed using methods such as subword segmentation (Kudo and Richardson, 2018; Sennrich et al., 2016c), augmentation by word-level alignment models and dictionaries (Arthur et al., 2016; Luong et al., 2015b), and pointer networks (Gulcehre et al., 2016; Minh-Cong et al., 2022; Vinyals et al., 2015). It is generally accepted that tokens seen more frequently in training are more accurately translated.

[4] https://github.com/neulab/compare-mt
[5] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/normalize-punctuation.perl
[6] https://spacy.io/models/en#en_core_web_trf

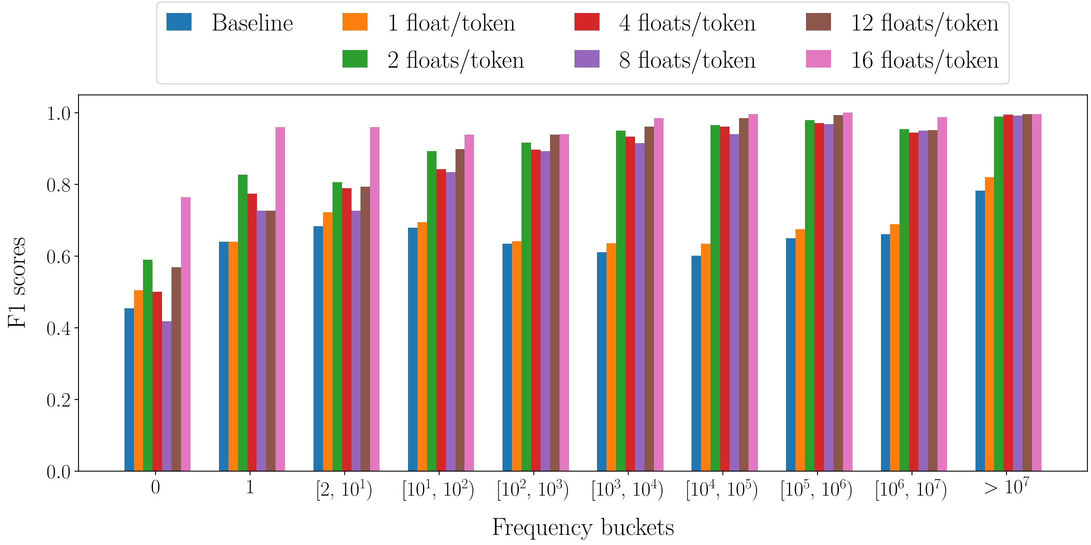

Figure 4.2: Frequency buckets vs. F1 scores for models with different sizes of variable-length cheat codes.

To verify this, we analyses the outputs from our cheat code models. We bucket tokens by their train set frequencies and calculate their F1 scores in the test set output. However, as evident from Figure 4.2, we find a different pattern:

- Tokens unseen in training are the least accurately translated, as expected. Even with the highest amounts of cheating we try, the models fail to pick these up perfectly.
- Tokens seen less than 100 times are translated relatively accurately. These are mostly names, which are often copied to the target correctly. In Table 4.2, the first example shows a name being omitted in translation, while the second shows it being copied correctly.
- Surprisingly, we see a slight drop in accuracy for tokens seen in the buckets between 100-100000 times in the baseline model and with lower levels of cheating, and the accuracy only catches up with the lower frequency buckets once they can cheat more. In some cases, this is due to the models paraphrasing words in these buckets more freely (see the third example in Table 4.2), since the words in this frequency range are usually content words and not function words (which might be relatively difficult to paraphrase) and thus they score lower on token-level matching. However, the fourth example in Table 4.2 shows that the translation being incorrect even after cheating can indicate an error in the source sentence – the word “Grenezn” is a typo, so the model is unable to generate the correct translation even with cheating.

- Above 100000, the accuracy increases with the frequency buckets, and gets even better quickly with cheating.

4.5.2 Token Accuracy by Part of Speech (PoS)

It is intuitively clear that different parts of speech are not equally difficult to translate. To identify the parts of speech that appear most challenging to translate with our models, we PoS tag the test set according to Petrov et al. (2012)'s tagset and measure word-level translation accuracy by part of speech. In this case, we show results on the fine-tuned models, but the trends are effectively identical for the cheat code models.

It can be seen in Figure 4.3 that verbs (label `VERB`) have the lowest accuracy in the baseline model — this is due to a lot of possible variation in conjugation, and so this quickly improves with cheating. Unknown words (label `X`) are also difficult for the model, as expected. Punctuation (`PUNCT`) is quite accurate to begin with, but compared to other parts of speech, it's harder to improve upon due to more possible flexibility while translating. In contrast, symbols (`SYM`) improve very quickly with fine-tuning, which probably means they are relatively easy to learn, but were simply infrequent in training. Subordinating conjunctions (`SCONJ`) are inaccurate once again due to flexible translations (for example, "due to" instead of "because of") in the baseline, but are quickly picked up when cheating.

By looking at Figure 4.3 at around 300 iterations, we can see that the models find verbs, adverbs, subordinating conjunctions, and auxiliaries hardest to learn.

4.5.3 Token Accuracy of Named Entities

Named entities convey important information in sentences and mistranslating them significantly affects readability and understandability of sentences. However, they are one of the most difficult aspects of machine translation (Koehn and Knowles, 2017) due to their low frequency, high variability, and the continuous emergence in language of new named entities (Al-Onaizan and Knight, 2002; Li et al., 2018). It is worth evaluating machine translation accuracy in detail across different categories of named entities to determine which ones are the most difficult to translate. We tag named entities in the test sets according to the OntoNotes 5.0 labels (Weischedel et al., 2013) and analyse the accuracy of each category.

Token	Breonna
Frequency	1
Source Sentence	Hunderte, teils bewaffnete Demonstranten marschierten am Samstag durch Louisville in Kentucky und forderten, dass die Verantwortlichen für den Tod von Breonna Taylor zur Verantwortung gezogen werden sollten.
Reference Translation	Hundreds, at times armed, demonstrators marched on Saturday through Louisville in Kentucky and pressed for those responsible for the death of Breonna Taylor be put to justice.
Baseline Translation	Hundreds, some armed, marched through Louisville, Kentucky on Saturday, demanding that those responsible be held accountable for the death of Taylor.
Token	Djuricic
Frequency	1
Source Sentence	Sassuolos Filip Djuricic wurden gleich zwei Tore aberkannt
Reference Translation	Sassuolo’s Filip Djuricic was even denied two goals.
Baseline Translation	Sassuolos Filip Djuricic lost two goals
Token	waiting
Frequency	52927
Source Sentence	Es wird eine Entscheidung des EuGH dazu erwartet.
Reference Translation	This is waiting on a decision from the EuGH.
Baseline Translation	A decision of the ECJ on this is expected.
Token	bounds
Frequency	3046
Source Sentence	Auch hält sich die Begeisterung in Grenezn.
Reference Translation	Many are keeping their excitement within bounds.
Baseline Translation	There is also enthusiasm in Grenezn.

Table 4.2: Examples of translations by the baseline model of words from different frequency buckets. Note that the last source sentence has a typo causing the untranslated word – see discussion in Section 4.5.1.

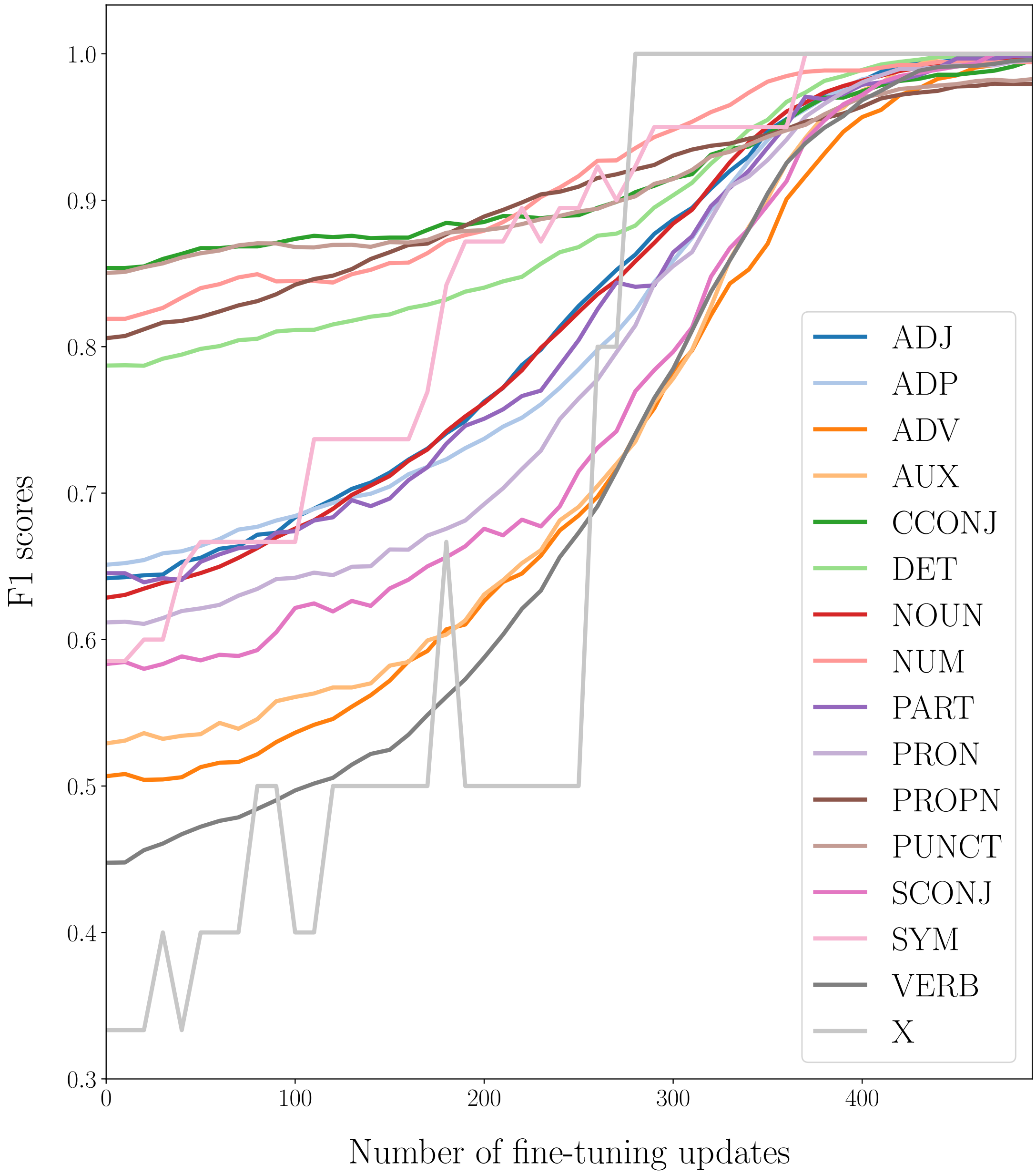

Figure 4.3: PoS F1 scores changing with fine-tuning on test set. 0 updates corresponds to the baseline model. Label `INTJ` is excluded because there were no instances in the test set.

Label	Baseline	cc1f	cc2f	cc4f	cc8f	cc12f	cc16f	cc25f
CARDINAL	0.763	0.785	0.776	0.798	0.824	0.829	0.884	0.910
DATE	0.816	0.816	0.823	0.838	0.837	0.848	0.879	0.914
EVENT	0.693	0.646	0.615	0.674	0.747	0.771	0.811	0.864
FACILITY	0.652	0.589	0.661	0.649	0.615	0.661	0.730	0.827
GPE	0.878	0.862	0.867	0.855	0.858	0.859	0.868	0.900
LOCATION	0.871	0.797	0.831	0.841	0.813	0.826	0.916	0.919
MONEY	0.675	0.650	0.564	0.633	0.667	0.675	0.649	0.938
NORP	0.753	0.772	0.748	0.782	0.760	0.790	0.831	0.885
ORDINAL	0.785	0.791	0.824	0.782	0.719	0.763	0.800	0.836
ORG	0.765	0.745	0.771	0.780	0.779	0.780	0.794	0.826
PERCENT	0.860	0.809	0.851	0.674	0.758	0.848	0.839	0.879
PERSON	0.885	0.890	0.890	0.883	0.892	0.883	0.877	0.890
PRODUCT	0.697	0.674	0.714	0.698	0.646	0.742	0.804	0.740
QUANTITY	0.648	0.615	0.621	0.662	0.712	0.667	0.727	0.841
TIME	0.679	0.643	0.660	0.660	0.683	0.706	0.761	0.838
WORK OF ART	0.607	0.585	0.565	0.571	0.555	0.699	0.703	0.593

Table 4.3: F1 scores of categories of named entities for different sizes of fixed-length cheat codes. ccNf indicates cheat codes of size N floats. Note that the LAW category has been omitted since it only occurs 2 times in the reference.

Table 4.3 shows the accuracies of different categories in detail for the baseline and the models with fixed-length cheat codes. Other types of cheating show similar results. The models find categories[7] like `PRODUCT`, `WORK OF ART`, and `GPE` relatively difficult to pick up with cheating, since these are relatively open-ended vocabulary classes. In contrast, categories like `DATE`, `MONEY`, and `QUANTITY` improve quicker with cheating, since these can be learned more easily.

Table 4.4 shows some examples of how the models get named entities wrong, and how they can reach the correct translation after a certain amount of cheating in some cases.

- The first example involving “Bayern” is quite difficult for the models due to the literal translation of “Bayern” to the literal “Bavarians” making the overall translation involving the football club “Bayern Munich”, referred to here as “the Bayern”, incorrect. The model learns to overcome this[8] with cheat codes of size 2 floats/token or after between 300 and 400 fine-tuning updates.

[7] Explanations of category labels can be found at `https://catalog.ldc.upenn.edu/docs/LDC2013T19/OntoNotes-Release-5.0.pdf#page=21`

[8] Note that the final translation is still incorrect due to the absence of a negation, but we still use this example to demonstrate the ability of the cheating method to pick up the word “Bayern”.

Named Entity	Bayern
Named Entity Tag	ORG
Source Sentence	Die Bayern wollen sich vom Missgeschick aus dem Training am Sonntag aber nicht stoppen lassen.
Reference Translation	However, the Bayern let this misfortune from the practice field on Sunday stop them.[10]
Baseline Translation	But the Bavarians do not want to be stopped by the mishap from the training on Sunday.
Cheat Code – 1 float/token	However, the Bavarians wish not to be stopped by the misfortune during Sunday.
Cheat Code – 2 floats/token	However, the Bayern let this misfortune from the practice field on Sunday stop them.
FT Iter. 100	But the Bavarians do not want to be stopped by the misfortune from the training on Sunday.
FT Iter. 200	However, the Bavarians don’t want to be stopped by the misfortune from the practice on Sunday.
FT Iter. 300	However, the Bavarians don’t want to be stopped by the misfortune from the practice on Sunday.
FT Iter. 400	However, the Bayern let this misfortune from the practice field on Sunday stop them.
Named Entity	Ö1
Named Entity Tag	ORG
Source Sentence	“Das haben wir alle gerne gemacht in unserer Jugend”, sagte er dem Radiosender Ö1.
Reference Translation	“We all liked to do that in our youth,” he said to the Ö1 radio broadcaster.
Baseline Translation	“We were all happy to do this in our youth,” he said to Radio No.1.
Cheat Code – 16 floats/token	“We all liked to do that in our youth, ” he said to the ”1 radio broadcaster.
FT Iter. 940	“We all liked to do that in our youth,” he said to the ’1 radio broadcaster.

Table 4.4: Examples of errors in named entity translations, and the change with increased cheating.

- The second example shows the name “Ö1”, which is never translated correctly, even with our highest levels of cheating, indicating that it’s very hard to translate for the models[9].

[9] This might ostensibly be due to the character Ö not occurring in English, but in fact it appears 342 times in the English training data.

4.5.4 The Fleetwood Mac Problem

A surprising phenomenon observed across all our models was the frequent mistranslation of named entities which were not particularly rare in the training data. One egregious example, shown in the first example in Table 4.5, is the name of the band "Fleetwood Mac", which appears 137 times in the English train set and correspondingly 135 times in the German source, but is repeatedly mistranslated not just by the baseline model, but also by the cheating models which score >90 BLEU overall on the test set. Another very prominent example is the city "A Coruña" (Table 4.5, third example), which occurs 822 times on the English side and 854 times on the German side in the training set, but does not get translated correctly a single time that it appears in the test set.

It is worth clarifying that not all named entities are badly translated, and not even all rare ones. For example, the name "Jürgen Mistol" never occurs in the training set and is translated correctly in the test set, while "Jürgen Klopp" occurs 99 times on the English side and 105 times on the German side in training, but is translated by our models as "Jürgen Lewandowski", "Juergen Murdoch", and "Jürgen Charlottesville" among other things (Table 4.5, fourth example). With the individual token "Mistol" appearing only 1 time in the training set (not preceded by "Jürgen") in contrast to "Klopp" appearing 362 times (English side), it is unclear why the models all struggle to translate the far more frequent name.

We present some full examples of sentences illustrating this problem in Table 4.5.

There are a few potential explanations for this phenomenon which we have considered, but fail to fully explain the phenomenon.

- Named entities can sometimes be segmented into long low-probability sequences of subwords, making them hard for models to generate. However, this does not seem to be the case based on some investigation – for example, "Jürgen" and "Klopp" are present in our subword vocabulary and are not segmented at all, so this does not explain why the model is unable to generate "Jürgen Klopp" in a translation given its presence in the source.
- There can be encoding issues with diacritics, or the absence of accented characters like ñ, ü, or Ö in the English dataset can make them difficult for the models to learn, but we verified that these are indeed present in the English training data and encoded correctly.

Name	Fleetwood Mac
Train Set Frequency	137
Source Sentence	Fleetwood-Mac-Mitgründer Peter Green gestorben
Reference Translation	Fleetwood Mac co-founder Peter Green has died
Baseline Translation	Co-founder Peter Green died
Cheat Code Model	Yankees Mac co-founder Peter Green has died
Fine-tuned Model	Lewandowski Mac co-founder Peter Green has died
Names	Greta Thunberg; Stephen Colbert
Train Set Frequency	69; 39
Source	Greta Thunberg war in der bekannten Latenight-Show von Stephen Colbert per Videoschalte zu Gast und verriet im Interview, was sie bei ihrer Begegnung mit Donald Trump im Kopf hatte.
Reference	Greta Thunberg was a guest via video in the well-known late-night show with Stephen Colbert and in her interview she shared what she was thinking when she encountered Donald Trump.
Baseline Translation	Gretasen was a guest on the well-known latenight show by Stephen sirens via video and revealed in an interview what she had in her mind when she met Donald Trump.
Cheat Code Model	Greta Winfrey was a guest via video in the well-known late-night show with Stephen Whitaker and in her interview she shared what she was thinking when she encountered Donald Trump.
Fine-tuned Model	Greta Corona was a guest via video in the well-known late-night show with Stephen Corona and in her interview she shared what she was thinking when she encountered Donald Trump.
Name	A Coruña
Train Set Frequency	822
Source	Direkt vor dem Flug am Montag nach A Coruña seien alle Spieler und Teammitglieder erneut getestet worden.
Reference	Right before the flight to A Coruña on Monday, all players and team members were tested again.
Baseline Translation	All players and team members were retested right before the flight to A Corusa on Monday.
Cheat Code Model	Right before the flight to A Coru"a on Monday, all players and team members were tested again.
Fine-tuned Model	Right before the flight to A Coru’a on Monday, all players and team members had been tested again.
Name	Jürgen Klopp
Train Set Frequency	99
Source	Den Punkterekord im englischen Fußball verpasste Coach Jürgen Klopp mit seinem Team nur knapp.
Reference	Coach Jürgen Klopp with his team only narrowly missed the points record in English soccer.
Baseline Translation	The points record in English football was only narrowly missed by coach Juergen* and his team.
Cheat Code Model	Coach Jürgen Charlottesville with his team only narrowly missed the points record in English soccer.
Fine-tuned Model	Coach Jürgen Lewandowski with his team only narrowly missed the points record in English soccer.

Table 4.5: The Fleetwood Mac problem: names seen many times in training still get mistranslated. Examples with the 16 floats/token (95.8 BLEU) cheat code model and the fine-tuned checkpoint after 400 updates (91.3 BLEU).
*Juergen instead of Jürgen is arguably a *correct* transliteration, but still strange considering Jürgen occurs more than 15x more frequently in training than Juergen.

- Multi-word named entities, especially names, can occur as combinations of otherwise common single word named entities in a variety of contexts, which can confuse the model during inference. For example, both “Fleetwood” and “Mac” occur frequently with other names and not just in the combination “Fleetwood Mac” in training, potentially leading to the model preferring to generate other higher-probabilty combinations. This could be mitigated by allowing the model vocabulary to include frequent multi-word named entities as a single token, which is currently not the case with our subword segmentation tools.

4.5.5 Comparison of Methods

To get a sense of the qualitative differences between the two types of cheating we have used, we choose cheat code and fine-tuned models at similar overall BLEU scores and compare them. The chosen models are shown in Table 4.6a.

We find that the fine-tuned models are significantly better than the cheat code models at translating rare words and named entities in the test set, because they are fine-tuned on the sentences containing the same words while the models with cheat codes did not observe them frequently while training and so is unable to capture them effectively in the cheat codes. When analysed by parts of speech (Table 4.6b), we observe that cheat codes are better at function words like particles, adpositions, determiners, etc. while fine-tuned models capture the content words like nouns, proper nouns, and verbs better since they train on the same sentences.

However, the overall evolution of accuracy remains largely the same between the two methods of cheating, as is additionally demonstrated by the first example in Table 4.4, where fine-tuning and cheat code models learn to translate “Bayern” correctly at approximately the same point of overall quality, i.e. at cheat codes of size 2 floats/token (92.4 BLEU) and after around 400 fine-tuning updates (91.3 BLEU).

	cc25f	iter300	cc2v	iter410
BLEU	67.0	67.9	92.4	92.3
NE	0.8713	0.9215	0.9664	0.9754

(a) Overall quality and accuracy on named entities.

Labels	cc25f	iter300	cc2v	iter410
ADJ	0.8123	0.8770	0.9703	0.9815
ADP	0.8716	0.8442	0.9914	0.9826
ADV	0.7651	0.7579	0.9731	0.9568
AUX	0.8355	0.7621	0.9663	0.9750
CCONJ	0.8819	0.9099	0.9719	0.9744
DET	0.9355	0.8950	0.9952	0.9890
NOUN	0.7859	0.8709	0.9670	0.9816
NUM	0.9001	0.9431	0.9828	0.9886
PART	0.8814	0.8419	0.9747	0.9789
PRON	0.8019	0.8431	0.9854	0.9805
PROPN	0.8469	0.9241	0.9494	0.9639
PUNCT	0.9043	0.9112	0.9277	0.9703
SCONJ	0.8399	0.7840	0.9914	0.9720
SYM	0.7222	0.9500	0.9268	1.0000
VERB	0.7265	0.7649	0.9604	0.9683
X	0.2222	1.0000	0.2857	1.0000

(b) Accuracy by PoS

Table 4.6: Comparison of two pairs of models with different cheating methods but similar overall performance. cc25f: Cheat code of size 25 floats. cc2v: Cheat code of size 2 floats per token. IterN: Fine-tuning checkpoint after N updates.

4.6 Conclusions and Future Work

In this chapter, we have used two methods of "cheating" to identify some harder problems for MT systems, and find that while very rare or unseen words are very difficult to translate, the accuracy of translation does not simply increase with frequency. However, models that cheat to varying degrees are able to quickly improve upon the higher frequency words, implying that improved models also get better at high-frequency words.

We identify certain parts of speech and categories of named entities that are difficult to translate, and also observe that even some high-frequency named entities are hard for these models to learn. The cause of this phenomenon remains unclear and is worth investigating in further detail in future work. Allowing subword vocabularies to represent frequent multi-word named entities as a single token could be a potential solution to this problem.

Additionally, we see that the presence of translation errors even after large amounts of cheating can indicate problems in the source sentence, rendering the model unable to translate it correctly. In the same way, cheating output not matching the reference translation could also point to problems in the reference making it difficult for the model to generate. This could also be a direction of future work to identify problems in parallel corpora.

The analyses presented in this chapter were all performed on a single language pair: German→English. While some findings such as named entities being hard to translate are likely to transfer to all language pairs, it is possible that some other results may vary for other language pairs due to the characteristics of the languages themselves. Similar analyses across more language pairs and models would be valuable to figure out how hard problems vary across languages, what the MT research community should focus on improving, and to provide a fine-grained glimpse into a possible future of MT quality through the lens of cheating.

Chapter 5

Document-Level MT with Large-Scale Public Parallel Corpora

In the previous chapters, we have explored the effect of leaking information from the target translation to an MT model to enable the model to capture additional information that allows it to produce a more accurate translation. While this is useful as an analysis tool, in reality, the target translation is obviously not available to us at inference. What we need is sources of additional information that are available to the model and mechanisms to incorporate this information into the translation process. In this chapter and the next, we explore two such scenarios.

One natural source of additional information in the translation process is context. When a human translator performs the task of translation, they usually do so for a full document or at least a paragraph of source language text. When translating a specific sentence, they therefore have access to the context in which the sentence is placed, which can influence and improve the translation significantly.

In this chapter, we investigate the ability of MT models to incorporate surrounding document context into translation. We start by creating a new large-scale dataset for document-level MT due to the scarcity of existing datasets of this kind. We then train context-aware MT models on this dataset, evaluate their translation quality relative to a standard sentence-level baseline, and further analyse the effect of varying amounts of context on the translation of targeted discourse phenomena.

Our document-level translation models, as expected, out-perform the sentence-level baseline, demonstrating the usefulness of contextual information in translation. We release our datasets and code to further enable what we hope is an ongoing transition to document-level MT research.

This content of this chapter is based on work published at ACL 2024 (Pal et al., 2024).

5.1 Motivation

Machine translation has traditionally been framed as a problem of translating source text to target text one sentence at a time. However, depending on the languages and the content of the text being translated, it is often the case that a sentence is impossible to translate well without further contextual information. Maruf et al. (2021) summarise several discourse phenomena that are impossible for sentence-level MT systems to deal with – including anaphoric pronouns, lexical cohesion, deixis, and ellipsis. As discussed in Section 1.1, there are also features like grammatical gender, number, style, and formality, that can sometimes not be determined from the individual sentence but are dependent on surrounding context. Therefore, it has been clear for many decades (Bar-Hillel, 1960) that an MT system cannot translate some sentences without the ability to capture other linguistic cues from context. Läubli et al. (2018) also showed that while sentence-level neural MT can appear high-quality out of context, human evaluators had a much stronger preference for human translation when evaluating translation at the document level.

Despite the fact that document-level MT has these inherent advantages over sentence-level MT, most machine translation systems continue to operate at a sentence level. There have been many efforts to incorporate document context into neural MT (Section 5.2). What almost all of these methods have in common is that they require parallel training data with document context. Our main contribution in this chapter is the creation of a truly large-scale public parallel corpus that has document context.

ParaCrawl (Bañón et al., 2020) produced large-scale parallel corpora and the released data includes information about the URLs from which the sentences were extracted, but the released corpora were only sentence-level. We use raw webpage text publicly available from ParaCrawl along with the officially released sentence-level corpora to assemble large-scale document-level parallel corpora for several language pairs. We release our code to generate the document-level datasets[1] as well as the datasets in five selected language pairs[2].

[1] `https://github.com/Proyag/ParaCrawl-Context`

[2] `https://huggingface.co/datasets/Proyag/paracrawl_context`. More language pairs may be released in the future; it requires a resource-intensive but simple process of running our code on any ParaCrawl language pair.

We then validate the usefulness of our datasets by training context-aware translation models for all of these language pairs, and find that models that are aware of target context perform better than sentence-level baselines, often in terms of overall translation quality, but more significantly when evaluated with respect to targeted discourse phenomena. We conduct rigorous evaluations of how the models use the available context to improve their translations. By varying the amount of preceding context available to these context-aware models at test time, we show that while information from the immediately previous sentence is most useful – as is intuitively obvious, longer-range context up to a certain extent can also help models translate phenomena like anaphoric pronouns more accurately.

5.2 Related Work

Many attempts to utilise document context in MT have introduced specialised architectures to encode context (Jean et al., 2017; Kuang et al., 2018; Maruf and Haffari, 2018; Miculicich et al., 2018; Tu et al., 2018; Voita et al., 2018) alongside source sentences.

Some methods like Junczys-Dowmunt (2019); Post and Junczys-Dowmunt (2023); Sun et al. (2022) eliminate sentence boundaries altogether and use a standard Transformer architecture to translate an entire chunk of text as a single sequence. However, unless they retain some sentence markers like in Junczys-Dowmunt (2019) or Tiedemann and Scherrer (2017), these models can be difficult to evaluate with our existing evaluation paradigms and metrics due to the dependence on sentence-level test sets. In those cases, we often have to rely on sentence-splitting heuristics and alignment methods just to be able to compute a sentence-level metric on the model outputs. As a result, our work chooses a method to translate with one sentence at a time as input, but with document context provided as a separate additional input. This allows the model to benefit from context information while still being simple to evaluate with existing metrics and test sets.

There are few existing parallel corpora of significant size that retain document metadata – examples are Europarl (Koehn, 2005) which had only around 2M sentences for the largest language pairs; CzEng (Bojar et al., 2016; Kocmi et al., 2020) containing 61M sentence pairs with document annotation along with more than 100M synthetic sentence pairs, but only for eng↔ces; OpenSubtitles (Lison and Tiedemann, 2016) with just under 50M sentence pairs with document annotations for the largest language pairs; and News Commentary (Kocmi et al., 2023). The latter two datasets are relatively large corpora for several language pairs, but are restricted in domain.

The CCAligned corpus (El-Kishky et al., 2020) includes hundreds of millions of comparable document pairs across many languages, from which sentence-level datasets were extracted and released. A dataset with sentence pairs and corresponding document contexts was not created; however, it should be possible to extract a similar dataset as the one we present here from the available data released by CCAligned which includes documents, URLs, and sentences.

In work concurrent and similar to ours, Wicks et al. (2024) also created document-level translation corpora by recovering context from the previously available EuroParl, News Commentary, and ParaCrawl datasets. Previously, Al Ghussin et al. (2023) used publicly available parallel document metadata from ParaCrawl[3] to extract aligned paragraphs of text, and used these paragraphs as a proxy for documents. Even though their extracted datasets were at a relatively small scale due to their use of only a subset of ParaCrawl data and strict filtering, they observed clear improvements in targeted evaluations of document-level translation phenomena.

Post and Junczys-Dowmunt (2023) showed that using only monolingual documents and back-translating (Sennrich et al., 2016b) them sentence by sentence to produce synthetic document pairs can surprisingly produce better results than using actual document pairs to train a document-level model. Their results, however, were mostly on unreleased private data, and their comparison could not be reproduced on public data precisely because of the absence of public datasets of adequate size.

Another recent orthogonal approach is to use LLMs' inherent ability to model long context to perform document-level translation (Karpinska and Iyyer, 2023; Wang et al., 2023; Zhang et al., 2023) with no or very few parallel training examples. While this paradigm is gaining popularity, it is yet to be comprehensively explored, and the need remains to have large-scale datasets of parallel text with document context.

[3] https://www.statmt.org/paracrawl-benchmarks/

Language pair	Sentences	Source	Target	Both
eng-deu	278.3	105.6	110.3	92.1
eng-fra	216.6	83.5	86.3	72.2
eng-ces	50.6	18.7	21.0	16.3
eng-pol	40.1	16.8	18.4	14.9
eng-rus	5.4	3.1	2.8	2.4

Table 5.1: Sizes of our document-level datasets in millions of lines. “Sentences” is the size of the original ParaCrawl sentence-level datasets. “Source/Target” denotes the subset of sentence pairs where there is at least one source/target context – eng is always considered the source language in this case. “Both” denotes the subset of sentence pairs with at least one source context and one target context. Note: the eng-rus dataset is significantly smaller because it was not part of the ParaCrawl main release, but a smaller “bonus” release.

5.3 Dataset

At the time of its release, ParaCrawl (Bañón et al., 2020) was the largest publicly available sentence-level parallel corpus for most of the languages it supported. The ParaCrawl corpus mining process included steps to match documents that were estimated to be translations of each other, from which sentences were extracted and aligned, but unfortunately, the released corpora did not preserve document context or structure, and only contained isolated sentence pairs along with the source URLs they were originally extracted from.

However, separately, a lot of the raw text crawled from the web was also released[4] as language-classified base64-encoded text with their corresponding URLs. Therefore, we were able to match the webpage contents to their URLs in the sentence-level parallel corpora to recover the corresponding documents.

To build document-level parallel datasets from these sources of data, we chose five language pairs – Czech (ces), Polish (pol), German (deu), French (fra), and Russian (rus), all paired with English (eng) – and used the following method:

1. Extract the source URLs and corresponding sentences from the TMX files from ParaCrawl release 9[5] (or the bonus release in the case of eng-rus). Each sentence is usually associated with many different source URLs, and we keep all of them.

[4] https://paracrawl.eu/moredata

[5] https://paracrawl.eu/releases

2. Match the extracted URLs with the URLs from all the raw text data and get the corresponding base64-encoded webpage/document, if available.
3. Decode the base64 documents and try to match the original sentence. If the sentence is not found in the document, discard the document. Otherwise, keep the 512 tokens preceding the sentence (where a token is anything separated by a space), replace line breaks with a special `<docline>` token, and store it as the document context. Since some very common sentences correspond to huge numbers of source URLs, we keep a maximum of 1000 unique contexts per sentence separated by a delimiter `|||` in the final dataset.
4. Finally, we compile three different files per language pair – a dataset with all sentence pairs where we have one or more source contexts, one with all sentence pairs with target contexts, and a third dataset with both contexts.

Even though the TMX files have source URLs for all released sentences, this process was lossy due to a few different reasons:

- ParaCrawl was compiled from a number of separate crawls or "collections", and there were inconsistencies in how URLs were formatted in intermediate steps. We employed some basic heuristics to match as many URLs as possible, such as removing `http://` and `https://` and trailing slashes before matching, but there is a still a chance some URLs were missed in this process.
- Data from CommonCrawl was not duplicated in the released raw text from ParaCrawl. To avoid re-downloading huge amounts of data, any URLs that were present in CommonCrawl but not in the other collections are missing from our dataset.
- There were many instances where the original sentence could not be found in the contents of a webpage corresponding to its source URL. This is most likely due to the same URLs being crawled at different times and finding dynamic or possibly entirely changed content.

Due to the existence of multiple matched documents for some sentences in the datasets, source and target contexts for a sentence pair may not be aligned. However, approximately 99.9% of all extracted sentence pairs have exactly one source/target context, which implies that the contexts should be aligned in most cases. Further filtering is recommended if aligned contexts are required, with the simplest option being to remove the subset of sentence pairs with more than one matched context.

The sizes of our extracted datasets are shown in Table 5.1, and can be seen to be significantly larger than any of the other publicly available document-level parallel corpora, as discussed in Section 5.2. Some samples can be found in Appendix A, and the full datasets are publicly available at https://huggingface.co/datasets/Proyag/paracrawl_context.

5.4 Document-level Translation Models

To evaluate the usefulness of our datasets, we train document-level translation models using only our datasets. Even though higher-quality document-level training data exists at a smaller scale, we choose to train our models only on data from ParaCrawl in order to accurately evaluate the quality and utility of our data and to make a fair comparison with our sentence-level baselines.

At training time, for each input example, we first sample one context out of up to 1000 that are present in the document-level dataset.

The amount of context available at test time is variable – there could even be none, and the amount of context useful or optimal to translate a sentence cannot be determined in advance. Therefore, to ensure that the models are capable of using variable-length context, we uniformly sample a context length l from $\{1,\ldots,256\}$ for each training example and retain at least l tokens from the preceding context, possibly exceeding the limit to avoid mid-sentence splits. This context sampling was implemented using a custom pipeline[6] in the Sotastream toolkit (Post et al., 2023). We then use the source sentence as the main model input, the sampled context as a second input (see detailed model architecture in Section 5.4.1), and the target sentence as the target model output.

We train separate models for each language pair using either source or target context information. While the model is always provided ground-truth context at training time, this is not always available in the case of target context at test time, so we test using both ground-truth target context and real predicted output context. However, we note that one of the most common use cases of machine translation is in the context of Computer-Assisted Translation (CAT) tools, where translators can see preceding context but typically machine translate and post-edit one sentence at a time, as a result of which gold-standard target context is available for each sentence.

[6] https://github.com/Proyag/sotastream/blob/custom_pipelines/sotastream/pipelines/sample_from_fields_pipeline.py

5.4.1 Model Architecture and Training

For all our models, we use the dual-encoder Transformer architecture from Junczys-Dowmunt and Grundkiewicz (2018) but without tied parameters between the two encoders, implemented in the Marian framework (Junczys-Dowmunt et al., 2018). In other words, we modify a standard Transformer encoder-decoder model (Vaswani et al., 2017), which takes the source sentence as input and produces the target sentence as output, to add a second encoder which takes additional source/target context as input. This is similar in spirit to Zhang et al. (2018), but we do not incorporate the context encoding into the source encoding, instead directly feeding both encodings to the decoder. As shown in Figure 2.2, the decoder has two stacked cross-attention sub-layers to attend to the two encoder representations. We use default Transformer-big hyperparameters. This choice of architecture is less complex than the specialised architectures described in Section 5.2, but still allows for separating the current sentence and context inputs, giving us greater control over evaluation and interpretability compared to models which translate an entire large chunk of text at a time. Moreover, the addition of the second encoder automatically accounts for the need for extra model capacity to encode the document context.

As discussed in Section 3.3.1, this dual-encoder architecture does not allow the second encoder access to the source sentence when encoding the context. This means that the model tries to capture all the information from the context in the second encoder, or ideally learns to extract what parts of context are most likely to be useful irrespective of the specific source sentence[7].

We train context-aware models for the following language pairs: eng→deu, eng→fra, eng→rus, eng→ces, and pol→eng.

We train our models with dynamic batch size to make optimal use of GPU memory. We train all models on 4 or 8 Nvidia A100 or 3090 GPUs, using gradient accumulation to ensure that the average effective batch sizes are approximately equivalent in each case. We validate every 50 million target tokens for eng→rus and every 500 million target tokens for all other language pairs, early stopping when cross-entropy calculated on the validation set does not improve for 10 consecutive validations.

[7] An alternative architecture could encode the context more efficiently by having access to the source sentence and therefore knowing exactly what information is required from the context. This was not explored in our work.

5.4.2 Test Data for Evaluation

To assess the translation quality of our models, we perform two kinds of evaluation – general MT quality metrics, and using contrastive evaluation to measure the accuracy of the models on targeted discourse phenomena.

General Translation Quality Metrics

We compute standard sentence-level quality metrics – BLEU (Papineni et al., 2002) using the sacreBLEU implementation[8] (Post, 2018) and COMET[9] (Rei et al., 2022) – on the following WMT test sets, all of which were released with document metadata: WMT22 eng→deu and eng→ces (Kocmi et al., 2022), WMT23 eng→rus (Kocmi et al., 2023), WMT20 pol→eng (Barrault et al., 2020), and WMT15 eng→fra (Bojar et al., 2015).

Contrastive Evaluation

Contrastive test sets consist of input text and translations which appear correct at the sentence level, but may be wrong given more context. Models are evaluated by their ability to assign higher probability to the sentences that are correct in context. We evaluate our models on a few different contrastive test sets which measure the following types of discourse phenomena for specific language pairs:

- **Anaphoric pronouns**: The ContraPro test sets for eng→deu (Müller et al., 2018) and eng→fra (Lopes et al., 2020) translation evaluate the accuracy of pronoun translation where the source English sentence does not contain enough information to determine the correct pronoun in the target language.
  The eng→deu test set contains examples of sentence pairs where the source English sentence contains the pronoun *it* which needs to be translated into one of *es*, *sie*, or *er* in the target German. The pairs are designed so that provided context information is required to determine the correct translation of the pronoun.
  A similar test set created by Lopes et al. (2020) evaluates the same phenomenon in eng→fra translation, where the English *it* is translated to *il* or *elle* in French and *they* is translated to *ils* or *elles*.

[8] `BLEU|#:1|c:mixed|e:no|tok:13a|s:exp|v:2.4.0`

[9] Specifically `wmt22-comet-da`

One part of the DiscEvalMT test set (Bawden et al., 2018) also evaluates anaphoric pronoun translation in eng→fra, containing examples where correctly generating a pronoun in the target French requires the model to use context information about the antecedent.

- **Deixis and ellipsis**: Good Translation Wrong in Context (GTWiC) (Voita et al., 2019) is a collection of contrastive test sets to evaluate a number of discourse phenomena in eng→rus translation, among which are deictic expressions, verb phrase ellipses, and correct inflection of nouns which depend on elided verbs.
  - **Deixis:** These examples are related to gender and formality marking in Russian that are absent in the source English. The model needs to use context to translate these deictic words or phrases correctly.
  - **Ellipsis:** These examples have elliptical constructions in the English text that cannot be elided in Russian, so the translation needs to expand the ellipsis. There are two kinds of ellipsis-related errors targeted here: where the target text has wrong morphological inflection due to missing information from the source ellipsis, and where the wrong verb is generated for a verb phrase ellipsis.
- **Lexical choice**: The DiscEvalMT contrastive test set from Bawden et al. (2018) tests lexical choice in eng→fra translation, where the test examples contain an ambiguous word in the source and the model needs to be able to use context information to disambiguate the correct sense of the word and translate it correctly. It also tests lexical cohesion, i.e. the ability of the model to ensure that named entities that are repeated in the source are translated consistently in the output. An eng→ces extension of the lexical cohesion subset was created by Jon (2019). GTWiC also has a similar subset to test lexical cohesion in eng→rus. Models need to be aware of preceding context to translate cross-sentential repetitions consistently.

5.5 Results and Analysis

We train sentence-level and document-level models in a few different configurations for comparison. Our baseline is a standard sentence-level Transformer-big model trained on all of the ParaCrawl parallel data for a given language pair. We also train sentence-level baselines on the subsets of sentence pairs for which source or target contexts could be extracted, thus ensuring a fair comparison in terms of the number and content of training examples. Finally, for each language pair, we train two different document-

Model	BLEU / COMET				
	eng→deu	eng→fra	eng→ces	eng→rus	pol→eng
Sentence-level	35.2 / 85.4	40.5 / 83.1	36.8 / 88.4	**22.8 / 75.4**	**33.5 / 83.7**
Subset - source	35.0 / 85.5	–	36.3 / 87.5	22.0 / 74.8	32.6 / 83.3
Subset - target	34.3 / 85.3	40.7 / 83.1	35.9 / 87.8	22.0 / 75.3	32.4 / 83.2
Source context	34.9 / 85.0	–	36.6 / 88.1	19.4 / 72.4	32.4 / 83.0
Gold target context	**37.4 / 85.9**	**42.6 / 83.2**	**37.3 / 88.5**	21.9 / **75.4**	32.8 / 83.3
Pred. target context	34.7 / 85.4	40.5 / 82.8	35.4 / 87.1	21.5 / 74.9	32.8 / 83.4

Table 5.2: Overall sentence-level BLEU/COMET scores on test sets for models in different configurations. Bold text highlights the highest score for each language pair. “Subset - source” and “Subset - target” are sentence-level baselines trained on the subsets of sentences that have source or target contexts respectively, i.e. the same number of training examples as the corresponding context-aware models. “Gold target context” and “Pred. target context” are the same model which encodes target-side context, but the latter uses the predictions from previous lines in the same document as its context instead of the ground-truth context.

level models: one which is aware of source context and one which uses target context. We further test the target context model in two different scenarios: using original ground-truth context for each test example and using the actual model output from previous sentences within a document as target context.

5.5.1 Effect on Overall Translation Quality

One of the ways we evaluate our document-translation models is simply in terms of overall sentence-level translation quality metrics. The results are summarised in Table 5.2.

We find that while source context does not seem to benefit overall translation quality[10], or at least not in a way that is reflected in these metrics, using the ground-truth target context generally improves translation quality over the baseline using the same number of training examples, i.e. “Subset - target” compared to “Gold target context” in Table 5.2. This corroborates the findings of Bawden et al. (2018) and Fernandes et al. (2021) that target context is more useful for translation than source context.

[10] We skipped training a source context-aware model for eng→fra due to their ineffectiveness across the other language pairs, explaining the gaps in Table 5.2

This improvement is not observed for eng→rus, probably due to the small number of training examples in the subset of sentences with target context not being enough to train a high-quality context-aware model. The sentence-level baseline using the full set of ParaCrawl sentence pairs ("Sentence-level") out-performs the context-aware models for the smaller language pairs, benefitting from having a much larger number of training examples.

An MT system in an environment where translated output is post-edited sentence by sentence, such as in CAT tools, has access to ground-truth target context for every line, and can thus benefit from the improved translation quality of the context-aware model. However, a fully automated document-level translation pipeline does not have this information available. To reproduce this scenario, we also try using actual model predictions from previous sentences within a document as target context ("Predicted target context" in Table 5.2), and find that this does not yield the same improvement as using ground-truth context. This could be explained as a manifestation of exposure bias (Ranzato et al., 2016) due to the models only being trained on ground-truth contexts and not being robust enough to accurately use the relatively noisy predicted context, resulting in errors being propagated through the context. In some cases, the difference may not even be an obvious error, but could instead be related to domain/style hints that are available in the original context to guide the translation but are lost in the context predicted by the model. While models can be made more robust against the propagation of errors through preceding context using methods like scheduled sampling (Bengio et al., 2015) to expose some generated context to the model during training, the loss of contextual hints is more difficult to remedy.

5.5.2 Accuracy on Contrastive Test Sets

We also perform evaluations on selected contrastive test sets for some language pairs, as mentioned in Section 5.4.2. Each example in a contrastive test set has a source sentence and source/target context along with two or more possible outputs, one of which is correct. We use our models to score all the possible outputs and say the model gets an example right if it assigns higher probability to the correct output than to the other options. We then calculate accuracy over the entire test set.

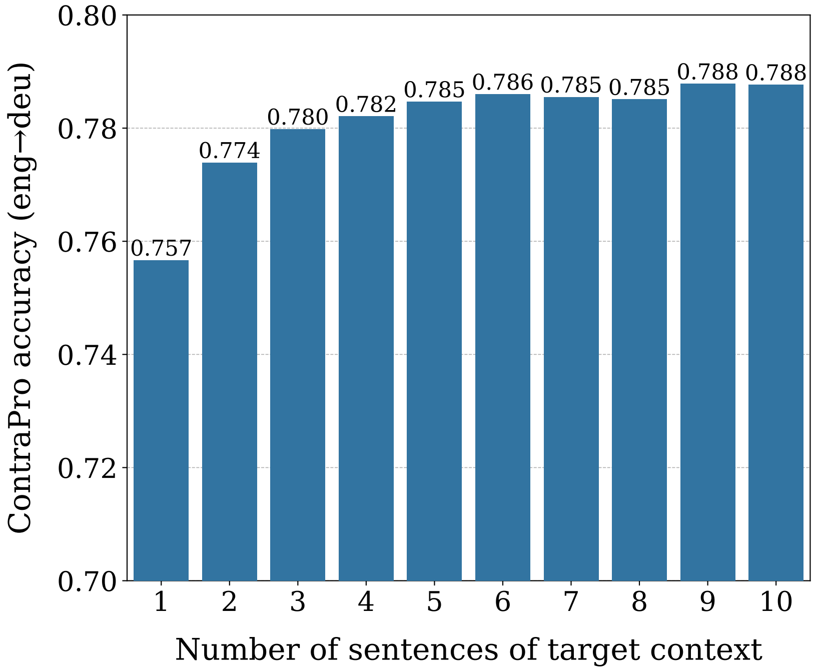

Figure 5.1: Effect of varying the number of lines of target context on ContraPro eng→deu pronoun translation accuracy. The baseline accuracy achieved by the sentence-level model is 0.507. Accuracy increases steadily with up to 3 or 4 sentences of target context with only marginal gains beyond that.

While the contrastive test sets include source context, we report results in this section only on our target context-aware models since, consistent with Table 5.2, we find that our source context-aware models are unable to outperform sentence-level baselines. Since each contrastive test set is different and designed for specific languages, we discuss them separately in this section.

ContraPro (eng→deu and eng→fra)

We use ContraPro (Müller et al., 2018) to evaluate the accuracy of pronoun translation for our eng→deu models. Each example has two translations which are both apparently correct at the sentence level but one of them uses the wrong pronoun in context. Our models are not required to generate translations, only to score each alternative output given the source sentence and ground-truth target context. If the model is able to take the context into account, it should assign higher probability to the option with the correct pronoun.

We find that while our eng→deu sentence-level model has an accuracy of 0.507, i.e. approximately random chance, the model with target context scores 0.785 when provided 5 sentences of preceding context. This shows that the model learns to accurately disambiguate the correct choice of pronouns using the context information.

For eng→fra, we find that the sentence-level model already achieves a reasonably high accuracy of 0.815, due to the fact that the antecedents of the pronouns are located within the same sentence in approximately 43% of the examples in this test set. Using our target context-aware model, we still see an increase in accuracy to 0.824 given a single sentence of preceding context, but no further improvement with longer context.

Effect of Context Size Figure 5.1 shows the effect of the amount of context that is exposed to the eng→deu model on its ability to accurately translate the anaphoric pronouns in ContraPro. We find that a single sentence of target context is enough to significantly increase the accuracy of pronoun translation beyond a sentence-level baseline's 0.507, and context longer than 3 sentences does not make much of a further difference in terms of total accuracy. This makes intuitive sense, since the antecedent of a pronoun is most often in the immediately preceding sentence and only very rarely more than 2 or 3 sentences away. However, for the subset of 442 examples (out of 12000) where the antecedent distance is greater than 3, the accuracy increases from 0.709 for the baseline to 0.908 for the context-aware model, which indicates that our model is in fact able to use long-range information to disambiguate pronouns.

Good Translation Wrong in Context (eng→rus)

We use the GTWiC test sets (see Section 5.4.2 for details) to evaluate our models' performance on deixis, ellipsis, and lexical cohesion in eng→rus.

Unlike ContraPro, contrastive examples in GTWiC have several incorrect translations and one correct translation for each given source sentence and context. A test example is considered correct if our translation model scores the correct translation higher than all of the incorrect translations. We report accuracies separately for each GTWiC test set evaluating different phenomena.

The performance of our target context-aware model on the GTWiC test sets is reported in Table 5.3. We observe that the context-aware model is significantly more capable of translating deictic expressions accurately. However, we find that it does not perform well on the ellipsis test sets, with verb phrase ellipsis accuracy surprisingly being worse than chance, and improvements on lexical cohesion are also marginal. This is possibly

Model / Context Length	GTWiC Accuracy			
	Deixis	Ellipsis		LC
		Infl.	VP	
Sentence-level	0.5	**0.5**	0.058	0.458
Trg context/1	0.586	**0.5**	0.07	0.468
Trg context/2	0.654	0.494	0.07	**0.472**
Trg context/3	**0.692**	**0.5**	**0.074**	**0.472**

Table 5.3: Accuracy of our target context-aware model on the GTWiC test sets with varying number of sentences of target context. "LC" denotes lexical cohesion. While performance on deictic expressions improves steadily with more context, lexical cohesion only improves very marginally, and verb phrase (VP) ellipsis accuracy remains very low.

Model	eng→fra	eng→ces
Lexical Choice:		
Sentence-level	0.5	0.5
Target context	**0.525**	**0.533**
Anaphora:		
Sentence-level	0.5	–
Target context	**0.545**	–

Table 5.4: Accuracy of target context-aware models compared to sentence-level models on the DiscEvalMT test sets in eng→fra and eng→ces. Context-aware models achieve higher accuracy in each case.

because the models need *both* source and target context to be able to model these phenomena accurately. For example, it is difficult for the model to be aware that an entity should be repeated if is not aware that both the preceding source and target contexts had occurrences of the same entity.

A maximum of 3 context sentences is available per example in GTWiC, and we once again find that having more target context can be useful for the model to translate deictic expressions correctly, but the benefits diminish as the model usually gets adequate context from the last one or two sentences.

DiscEvalMT (eng→fra and eng→ces)

The DiscEvalMT eng→fra contrastive test sets (Bawden et al., 2018) evaluate two document-level translation phenomena: anaphora and lexical choice (see Section 5.4.2 for more details). The eng→ces extension of DiscEvalMT (Jon, 2019) only includes the lexical choice test set.

These test sets also have two alternative translations for each input sentence and context, and our models are expected to score the correct option in context higher than the other option.

We can see in Table 5.4 that while our document-level models do not score very highly on this benchmark, they still out-perform the sentence-level baseline. Similar to GTWiC, for the lexical cohesion test sets, we believe that the models would perform better if they were able to access both source and target contexts, since they are otherwise unaware of the repeated word or phrase.

5.6 Conclusions

In this chapter, we have described the construction and release of large-scale document-level parallel corpora in five language pairs extracted from the ParaCrawl datasets in an effort to mitigate the dearth of publicly available MT training data with document context. We also open-source code to enable the community to compile such datasets in more language pairs. Due to both the ParaCrawl pipeline and our code being open-source, it is possible for the community to create document-level datasets for any supported language by crawling the web, and not just those already released by ParaCrawl.

While we treat any preceding text at the same URL as “context”, it is often the case that these are completely unrelated to a given sentence, such as UI elements, boilerplate text, or entirely unrelated content on the same webpage. Future work should also explore filtering these datasets to retain only genuine contextual information, which is likely to be much more useful to the model, although even content that is not strictly document context may help guide translation through indirect domain or style cues.

Our document-level translation experiments show that sentence-level models enhanced with target context improve in terms of overall translation quality as well as in terms of some targeted discourse phenomena compared to a purely sentence-level baseline. We show that MT can benefit from multiple sentences of preceding context to accurately translate discourse phenomena like anaphoric pronouns, although very long-range context is rarely useful.

This is consistent with our finding from Chapter 3 that the amount of information that needs to be added to sentence-level models is not very large, but adding the right information enhances the quality of translation.

To quantify the amount of information that context-aware models need from document context, future work could also use the cheat codes method from Chapter 3 to bottleneck the context representations and observe the effect on translation quality. This would enable us to analyse how much context[11] is really necessary and optimise the representation of context to ultimately make more efficient document-level translation models.

5.7 Limitations

While we hope that our released datasets are a valuable resource for the community to explore document-level MT, there are some limitations that should be considered when using these datasets that we briefly discuss below.

Relevance of context

Our work assumes that any extracted text preceding a given sentence on a webpage is relevant “document context” for that sentence. However, it is likely in many cases that the extracted context is unrelated to the sentence, since most webpages are not formatted as a coherent “document”. As a result, the dataset often includes irrelevant context like lists of products, UI elements, or video titles extracted from webpages which will not be directly helpful to document-level translation models.

Unaligned contexts

For sentences with multiple matching contexts, the source and target contexts may not always be aligned. However, as mentioned in Section 5.3, the vast majority of sentence pairs have exactly one source/target context, and should therefore have aligned contexts. We recommend filtering on this basis if aligned contexts are required.

[11] In terms of encoded size, not number of tokens.

Availability of both contexts

Our models are all trained with either source or target context being available to the models, but for some document-level phenomena like lexical consistency of repeated named entities, it is probably necessary for the model to be aware of both source and target context. Our datasets make it possible for future work to extract training data with both contexts and train such models.

Model quality

Our models are trained only on noisy ParaCrawl data and tested on high-quality WMT data. While there are much smaller but relatively high-quality document-level training datasets available, all our experiments were conducted only on ParaCrawl data to test the quality of the datasets without being influenced by other data. As a result, these models are not necessarily the strongest possible translation models, but they are useful to fairly and clearly compare document-level MT against sentence-level models.

Language coverage

ParaCrawl was focused on European Union languages with only a few "bonus" releases for other languages. Moreover, most of the corpora were for English-centric language pairs. Due to the high computational requirements to extract these corpora, our work further chose only a subset of these languages, resulting in corpora for only a few European languages, some of them closely related. Given the availability of raw data and tools to extract such corpora for many more languages from all over the world, we hope the community is encouraged to build such resources for a much larger variety of language pairs. Document-level translation phenomena also vary widely by source and target language, so such experiments for more languages is left for future work.

5.8 Harmful Content

The main released corpora from ParaCrawl were filtered to remove sensitive content, particularly pornography. Due to pornographic websites typically containing large amounts of machine translated text, this filtering also improved the quality of the resulting corpora. However, when we match sentences with their source URLs, it often happens that an innocuous sentence was extracted from a webpage with harmful content, and this content is present in our document contexts. This is a safety concern and we may release filtered versions of these corpora in the future, pending further work to filter harmful content at the document level.

Chapter 6

Improving Isochronous MT for Automatic Dubbing

In this final chapter, we explore an application of MT where external constraints are imposed on the translation, and the MT model needs to be aware of and obey these constraints alongside the input text in order to produce appropriate translations. We look at automatic dubbing, where the task is to translate the audio from video content into another language while attempting to maintain synchronisation between the target language audio and the original video so that the dubbed video looks natural.

So far we have studied MT in settings where the only factor affecting translation is the content of the source and target text and their surrounding textual context. The automatic dubbing scenario introduces us to a new kind of external constraint: the timing of the spoken content. This brings additional challenges to these MT systems, since the translations need to not only be *correct*, but also fit these timing restrictions.

In this chapter, we use target factors in a Transformer model to inject timing information into the model, and enable it to predict durations jointly with target language phoneme sequences. We also introduce auxiliary counters to help the decoder to keep track of the additional timing information while generating target phonemes. We show that our model, when provided segment duration information alongside source text, improves isochrony and the overall quality of the final dubbed output compared to isochrony-unaware models. It also improves upon previous work where the translation model is instead trained to predict interleaved sequences of phonemes and durations. Our work thus shows the benefit of modelling timing constraints jointly with the sentence-level translation task in this scenario.

The content of this chapter is mainly based on work published at Interspeech 2023 (Pal et al., 2023). The models built through this work were used as a strong baseline in the automatic dubbing shared task at IWSLT 2023[1] (Agarwal et al., 2023) and IWSLT 2024[2] (Ahmad et al., 2024), through which we obtained human evaluation results, which are also presented in this chapter.

6.1 Motivation

Automatic dubbing (Federico et al., 2020a) aims to translate speech from video content (such as movies and TV shows) into a target language while maintaining isochrony, i.e. matching the speech and pause structure of the source speech in order to preserve time synchronisation in the dubbed video. In the standard automatic dubbing pipeline, an Automatic Speech Recognition (ASR) system transcribes the source audio into source language text, the text is translated into the target language by a Machine Translation (MT) system, after which a Prosodic Alignment (PA) module inserts pauses to segment the translated text, before a Text-to-Speech (TTS) system generates target language speech (see Section 6.2 for a more detailed description of the pipeline).

A major drawback of this pipeline is the fact that since the MT system is unaware of isochrony constraints, it can generate translations which do not fit the timing of the source audio. After segmenting target text through the PA module, to ensure the segments fit the speech timing, the speaking rate has to be adjusted for the TTS system, often resulting in unnatural output speech.

To translate speech for the purpose of automatic dubbing, translation thus needs to be isochronous, i.e. translated speech segments need to be aligned with the source in terms of speech durations. Simply generating a translation based on the transcribed source text is unlikely to be sufficient, as it highly probable that the translated audio will then be unsynchronised with the video, or appear/sound unnatural if forcibly aligned with video. The MT system therefore needs to take timing information as additional input alongside the source text and incorporate this information into the translation process.

[1] https://iwslt.org/2023/dubbing
[2] https://iwslt.org/2024/dubbing

Our goal is to make the MT model aware of timing constraints and jointly optimise translation quality and isochrony, i.e. predict translations and target-side timing information using the same model to generate translations of high quality while matching the source's speech timing. We achieve this using target factors (García-Martínez et al., 2016), where alongside predicting phoneme sequences as the target, we also predict durations for each phoneme as a target factor. Additionally, we design *auxiliary counters*, which are modified target factors providing additional information to the decoder to help the model keep track of timing but whose outputs we do not use (these are described in detail in Section 6.3.1). Our main contributions in this chapter are thus the following:

- We show that target factors can be adapted to predict durations alongside phoneme sequences to jointly optimise translation quality and speech overlap for automatic dubbing.
- We design auxiliary counters to provide extra information to the model to keep track of timing information that further improves the speech overlap.
- We evaluate our models through automatic metrics and human evaluation, both of which show that our approach improves upon previous work which instead proposed a model generating interleaved sequences of phonemes and corresponding durations.
- We release our implementation[3] and scripts[4] sufficient for replication, to enable future research in this area.

6.2 Automatic Dubbing

Standard automatic dubbing methods (Federico et al., 2020a) usually use a complex pipeline (Figure 6.1(a)) involving several steps:

1. **ASR:** The source speech is transcribed into text using an ASR system.
2. **MT:** The transcribed text is machine translated into the target language.
3. **PA:** The translated text is segmented into phrases and pauses matching the source speech timing using a Prosodic Alignment (PA) module (Federico et al., 2020b; Virkar et al., 2021,2; Öktem et al., 2019).

[3] https://github.com/awslabs/sockeye/pull/1082

[4] https://github.com/amazon-science/iwslt-autodub-task

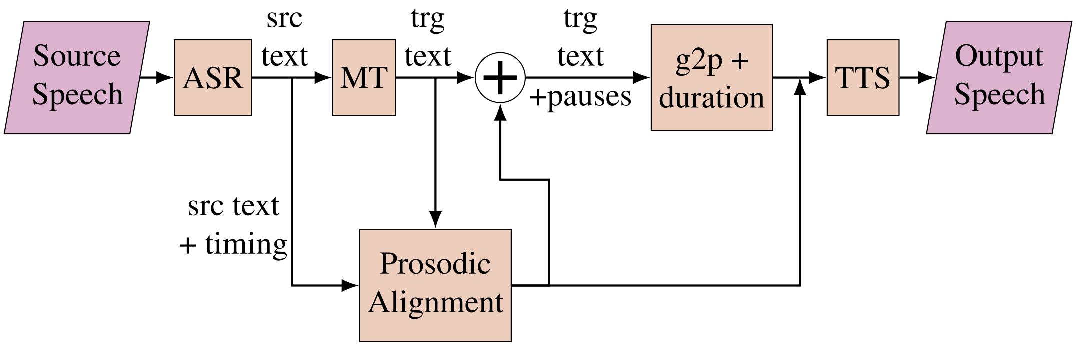

Figure 6.1(a): Standard automatic dubbing pipeline, with separate MT, PA, grapheme-to-phoneme, and duration prediction components.

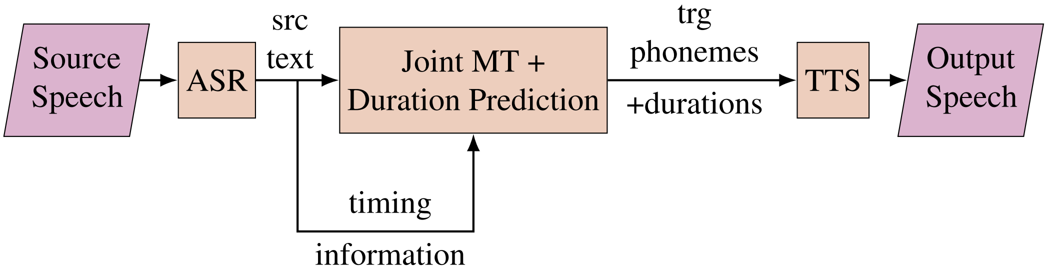

Figure 6.1(b): Modified automatic dubbing pipeline, with an isochronous MT model.

4. **Grapheme-to-phoneme conversion & duration prediction:** The translated and segmented text is converted into phonemes and the durations of the phonemes are predicted.
5. **TTS:** The phonemes and durations are fed to a TTS system to synthesise speech in the target language.

Since the MT module in the pipeline is unaware of any timing information, the produced translation, after the insertion of pauses, needs to be stretched by the TTS module to fit the source audio timing. This can result in the output speech sounding unnatural, with drastic changes in speaking rate within a sequence to satisfy the timing constraints of each segment. Some prior works have tried to avoid the separate PA step by training models to predict pauses within translations (Tam et al., 2022), integrating isochrony constraints in MT decoding (Saboo and Baumann, 2019) or by optimising prosody jointly with the TTS (Hu et al., 2021). An isochronous MT module that jointly models translation and duration enables the generation of translations more suited to the audio timing while simultaneously simplifying the pipeline (Figure 6.1(b)).

As a proxy for isochrony, some prior works have proposed optimising isometry, i.e. generating translations which match the number of characters in the source text (Lakew et al., 2021,2), but this has been shown to be weakly correlated to isochrony (Brannon et al., 2023).

Work concurrent to ours (Wu et al., 2023) predicted word durations along with words and presented a novel loss function for decoding. Chronopoulou et al. (2023) presented a simple sequence-to-sequence approach to generate interleaved sequences of phonemes and corresponding durations. We follow this data and model setup and use their approach as our baseline.

6.3 Method

We propose predicting phoneme durations as target factors (García-Martínez et al., 2016) (see Section 2.2.3), instead of generating interleaved sequences of phonemes and phoneme durations (Chronopoulou et al., 2023), and providing timing information to the model through a combination of duration pseudo-tokens and auxiliary counters (Section 6.3.1).

Target factors are additional output layers to produce multiple outputs at each decoder step. There are separate embedding layers for each target factor, and all the factor embeddings are concatenated to the main target embedding and provided as input to the decoder. To condition the factor outputs upon the main output, factors are shifted such that the factors corresponding to output token y_t are generated at step $t+1$. We use the Sockeye[5] target factor implementation.

In contrast to the interleaved baseline (Chronopoulou et al., 2023), target factors allow us to model the phonemes and durations separately while still ensuring that they are conditioned on each other. It also significantly decreases the sequence length and eliminates the possibility of producing invalid interleaved output (for example, two duration tokens in a row).

6.3.1 Auxiliary Counters

In addition to the main output f^{main} and corresponding durations being generated as a target factor (f^{dur}), we propose additional input embeddings in the decoder to help the model keep track of the isochrony constraints, which we denote auxiliary counters. The counters are implemented identically to target factors (i.e. each counter has an embedding layer whose embeddings are concatenated to the target embedding), except that the counters are not predicted at inference. Instead, the values of the counters are calculated at each time step based on the prior durations predicted by the model and used as input in the next step. These counters are:

- **Total frames remaining** (f_t^{total}): Keeps track of the total number of frames remaining in the sentence. This is initialised by the total desired duration of the sentence and is decremented by the phoneme duration at each output step.

$$f_t^{\text{total}} = f_{t-1}^{\text{total}} - f_t^{\text{dur}} \tag{6.1}$$

- **Pauses remaining** (f_t^{pause}): Keeps track of the number of pauses remaining in the sentence.

$$f_t^{\text{pause}} = \begin{cases} f_{t-1}^{\text{pause}} - 1, & \text{if } f_t^{\text{main}} = [\text{pause}] \\ f_{t-1}^{\text{pause}}, & \text{otherwise} \end{cases} \tag{6.2}$$

[5] https://github.com/awslabs/sockeye

Target text		you			know			[pause]	it		
f^{main}	NULL	Y	UW1	$\langle$eow$\rangle$	N	OW1	$\langle$eow$\rangle$	[pause]	IH0	T	$\langle$eow$\rangle$
f^{dur}	NULL	3	7	0	5	41	0	0	5	7	0
f^{total}	68	65	58	58	53	12	12	12	7	0	0
f^{pause}	1	1	1	1	1	1	1	0	0	0	0
f^{segment}	56	53	46	46	41	0	0	12	7	0	0

Table 6.1: An example target phoneme sequence along with its target factors. The corresponding word sequence is shown in the first row but not used in the actual model. The factors are time-shifted internally to condition factor outputs on the main output, which is not shown here. NULL is a padding token used to align the tokens correctly after the internal shift, so that the model sees f_0^{total}, f_0^{pause}, and f_0^{segment} before generating the first phoneme and duration.

- **Segment frames remaining** (f_t^{segment}): Keeps track of the number of frames remaining in a segment, i.e. until a pause is generated, or the sentence ends. This is initialised by the segment durations from the source sentence, and is decremented by the phoneme duration at each step until a `[pause]` is generated.

$$f_t^{\text{segment}} = \begin{cases} f_{t-1}^{\text{segment}} - f_t^{\text{dur}}, & \text{if } f_t^{\text{main}} \neq \text{[pause]} \\ \text{next segment duration}, & \text{otherwise} \end{cases} \tag{6.3}$$

All of these auxiliary counters are calculated from the phonemes and durations in preprocessing for training, and calculated at each time step at inference time. While the model can generate predictions for counters as target factors, we only use the counters to help the model keep track of its state and discard their outputs.[6] An example of a target sequence along with its target factor and counters is shown in Table 6.1.

The implemented behaviour of target factored models in Sockeye at inference time is to predict target factors and then feed those predictions back into the model at the next inference step. For counters (where we are trying to help the model keep track of timing), we found it critical to correctly calculate counter values according to the equations in Section 6.3.1 before feeding them back to the decoder at the next time step. Compared to the default Sockeye behaviour for target factors, this improved speech overlap significantly (from 0.9181 to 0.9972) without affecting translation quality.

[6] Additionally, our best models are trained without any gradient coming from the auxiliary counter predictions, effectively removing the part of the network predicting auxiliary counter outputs.

6.3.2 Noising Segment Durations

We show in Section 6.5 that our method is able to satisfy the duration constraints almost perfectly while maintaining reasonable translation accuracy. However, in practice we do not want to achieve perfect speech overlap because it can result in poor translations or speech that is shortened/lengthened to the point where it sounds *unnatural*. In fact, analysis of human dubbing (Brannon et al., 2023) has shown that human dubbers prioritise naturalness and translation quality over speech overlap, and median overlap is just 0.731 in a large corpus of human dubs.

For this reason, to relax the timing constraints in a controllable manner, we add Gaussian noise to the segment durations in our training data. This creates training examples where part or all of the translation ends slightly before or after the counters reach zero, and the model learns to be flexible with the timing information. We control the amount of noise by varying the standard deviation σ of the Gaussian noise.

An alternative approach would be to modify the loss function during training to reduce the effect of mismatched duration outputs relative to the main phoneme outputs. In fact, it should be possible to derive a modified loss function[7] that would be equivalent to adding noise to the segment durations. However, we choose to use the Gaussian noise method since prior work in isometric MT (Wilken and Matusov, 2022) and automatic dubbing (Chronopoulou et al., 2023) has shown it to be effective, and additionally, implementation is reduced to a simple pre-processing step as opposed to modifying the loss function.

6.4 Experiments

6.4.1 Dataset

We use the English-German subset of CoVoST-2[8] (Wang et al., 2021a) as our dataset, consisting of English audio clips and transcripts along with German text translations. Each clip roughly corresponds to a sentence. We run the Montreal Forced Aligner (MFA) (McAuliffe et al., 2017) on the English audio and transcripts to get sequences of phonemes with corresponding durations. This sequence becomes our target and the German transcripts are used as the source. The translations in the original CoVoST-2

[7] We do not attempt to derive this since we have not used such a modified loss in this thesis.
[8] https://github.com/facebookresearch/covost

corpus were done directly from the English transcripts and not with isochrony in mind, so our source (German) sentences do not necessarily fit the speech timing. This may result in sentence pairs where the two sides are significantly different in spoken duration, which is not the same scenario as real dubbing data. However, this setup still enables us to evaluate the ability of our models to generate translations that fit a given timing pattern, even if that pattern does not originate from the actual source sentence, which is the essential challenge in automatic dubbing.

We mark silence of more than 0.3 seconds with `[pause]` tokens in the target phoneme sequence in order to be able to reinsert these periods of silence in final dubs. We also mark the end of words with `<eow>` tags.

We calculate the duration of each segment (speech without pauses) by adding the phoneme durations between pauses. Since using the raw durations in seconds/frames would result in sparsity in training data, we bin the durations into 100 bins of approximately equal frequency. To add this speech duration information to each source sentence, we then concatenate tags of the form `<binN>` (where bin N represents the Nth of 100 bins) containing the binned duration of each corresponding speech segment. We apply Byte Pair Encoding (BPE) (Sennrich et al., 2016c) on the German text with 10k merges.

Our final dataset consists of 289,074 training examples, with 15,499 examples in the validation set and 15,413 in the test set.

6.4.2 Model Architecture

For all models, we use a standard Transformer-base architecture (Vaswani et al., 2017), augmented with target factors and counters where applicable, trained with a maximum batch size of 32768 tokens for 600 epochs, with a dropout probability of 0.3 and label smoothing 0.1. We save checkpoints every 2000 updates and pick the best checkpoint according to COMET scores on the validation set.

6.4.3 Baselines

Our main isochronous baseline follows the approach described by Chronopoulou et al. (2023), which is a simple Transformer sequence-to-sequence model. The input is the subword-level source text, with binned segment durations appended as tags to the end of the sequence and the output sequence is an interleaved sequence of phonemes and corresponding durations, with `<eow>` tags to mark the end of each word and `[pause]` tokens to mark the end of a segment.

For example, a source sentence with its corresponding target sequence is formatted as:

```
Source: Das weißt du nich@@ t@@ ? <||> <bin4> <bin1>
Target: D 2 OW1 5 N 6 T 8 <eow> Y 3 UW1 7 <eow> N 5 OW1 41 <eow>
[pause] IH0 5 T 7 <eow>
```

Additionally, we train a German→English MT model using the same datasets at the subword level (instead of phoneme outputs), and a model to translate German text to English phoneme sequences without durations. These two models act as baselines to measure how much the translation quality deteriorates for models with duration constraints.

6.4.4 Evaluation

Since our models output sequences of phonemes, we train a Transformer sequence-to-sequence model on the same dataset to transform English phoneme sequences into sequences of English words. Translation quality is then evaluated using BLEU[9] (Papineni et al., 2002; Post, 2018), Prism (Thompson and Post, 2020a,2), and COMET[10] (Rei et al., 2020). We find the metrics to be highly correlated in our results, and thus report only BLEU scores.

To quantify speech overlap between the reference (ref.) and the hypothesis (hyp.), we use the relative difference of duration between reference segments and predicted translated segments, averaged over all segments in the dataset:

$$\text{Speech Overlap} = 1 - \frac{|\text{ref. duration} - \text{hyp. duration}|}{\text{ref. duration}} \tag{6.4}$$

As an additional automatic metric, we also count the number of sentences in the validation and test sets where the wrong number of pauses is generated, since matching the timing of pauses in speech is particularly important to preserve lip synchronisation with the original videos.

[9] SacreBLEU: `BLEU|#1|c:lc|e:no|tok:none|s:exp|v:2.3.1`

[10] `wmt20-comet-da`

Model Configuration	BLEU ↑	Speech Overlap ↑
Text to text (MT)	38.0	–
Text to phonemes	35.8	–
Interleaved, no noise	32.0	0.8702
Interleaved, noised σ=0.2	35.4	0.7105
Single target factor, no noise	33.8	0.8931
+ all counters, no noise	34.0	0.9887
+ all counters, noised σ=0.1	35.6	0.8649

Table 6.2: Summary of key results for some representative models on the test set. The models with all the target counters use the optimal configuration from Section 6.5.

6.5 Results and Analysis

We train models to predict phoneme durations as a target factor and use all the auxiliary counters described in Section 6.3.1. Table 6.2 shows the results of some representative models.

The translation quality of the text-to-phones baseline is 2.2 BLEU lower than the text-to-text (i.e. standard MT) model (see Table 6.2). This is likely due to the following reasons:

1. To evaluate the text-to-phoneme model, we are mapping phonemes to words using a sequence-to-sequence model and then scoring with word-level metrics. The phonemes-to-words model is not perfect and is likely to be introducing some errors, making the text-to-phoneme model appear to be worse than it actually is.
2. We did not attempt to optimise the Transformer hyperparameters for phonemes, instead simply using the same hyperparameters as the standard MT baseline.

We find that modelling phoneme durations using target factors improves both translation quality (+1.8 BLEU) and speech overlap (+0.023) relative to the interleaved baseline (see Table 6.2, no noise settings). We note that due to the additional factor embedding and output layers, the factored models have a larger total number of parameters than the interleaved baseline. Since the rest of the model architecture is identical, we consider this a fair comparison for the sake of evaluating our modified model.

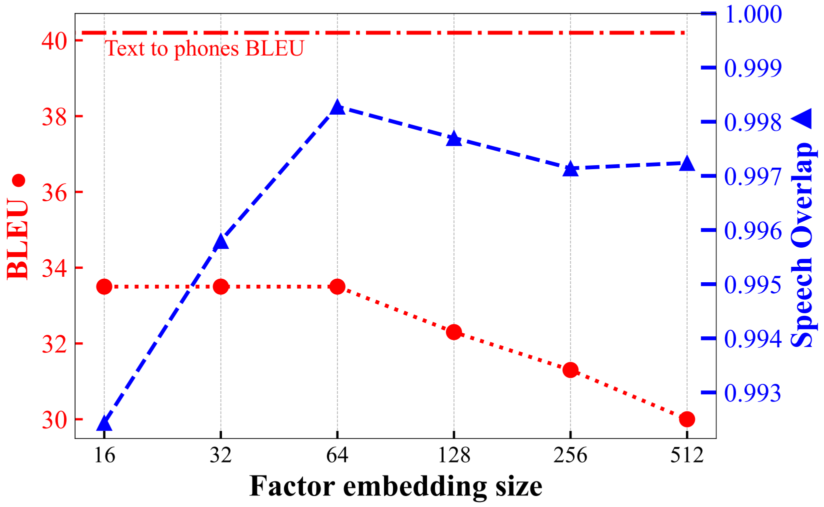

Figure 6.2: Results on the validation set varying the factor and counter embedding sizes. f^{pause} embedding size is always half of the other counters. Models trained with equal loss weights on factors and counters, on data with clean segment durations.

Adding auxiliary counters provides nearly perfect speech overlap (0.9887, perfect score is 1.0) in the no noise setting. It provides substantial improvement in speech overlap (+0.0956) compared to the target factor model without auxiliary counters, while marginally improving translation quality (+0.2 BLEU) (see Table 6.2, no noise settings).

By adding noise to the speech segment durations, we are able to obtain nearly the same translation quality as the text-to-phoneme model (35.6 vs 35.8) while still achieving very high speech overlap (0.8649, higher than observed in human dubs).

Embedding Size

Since the factored architecture adds a large number of parameters in the form of embedding matrices and output layers to the model, we want to optimise the factor embedding size so that it is large enough to adequately represent all the possible factor/counter values while not being too large to train in our limited data scenario. We sweep through a range of embedding sizes (Figure 6.2) and find that 64 dimensions is an optimal size. We set the embedding size for the f^{pause} counter to half of the other counter embeddings since it has much fewer possible values than the other counters.

Counter Loss Weights

At training time, counters are predicted at each step just like target factors since they are implemented identically, and all the factors/counters are assigned an equal weight for loss computation by default, i.e. the cross-entropy losses for the output and all

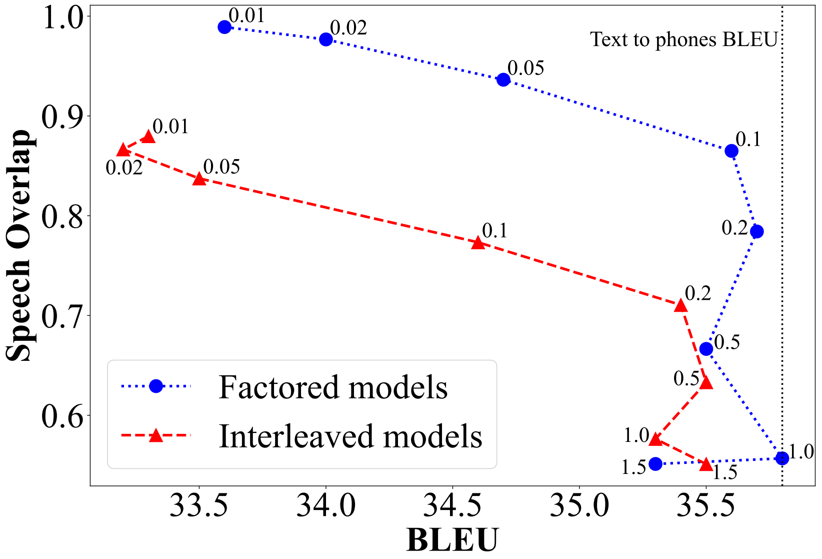

Figure 6.3: Variation of translation quality (BLEU) and speech overlap with different amounts of noise added to the segment durations. Results shown on the test set. Each point annotation indicates the standard deviation σ of the added noise.

target factors are simply summed. However, it is possible to generalise the loss by assigning different weights to the outputs. Since we do not use the outputs for the counters, we can set the loss weights of the counters to 0, thus letting the model focus on the phoneme and duration outputs that we actually need. We find that zeroing the loss weights of the counters helps improve translation quality by 4.2 BLEU at the cost of a very small drop (0.003) of speech overlap.

Adding Noise

We find that as we add more noise by increasing the standard deviation of the Gaussian noise (Section 6.3.2), for both the interleaved as well as factored models, the translation quality increases at the cost of speech overlap, ultimately matching the text-to-phonemes baseline (Figure 6.3). The amount of noise allows us to control the trade-off between translation quality and speech overlap.

However, the primary motivation of adding noise was to avoid the output speech sounding unnatural, which the outputs with perfect overlap often do (Section 6.6). Human evaluation results in Section 6.6 also support the hypothesis that adding some noise to the timing results in more natural dubbed output.

Model configuration	BLEU ↑	Overlap ↑	#Pause Mismatch ↓
All counters + source durations	34.0	0.9887	29
Without:			
Source durations	33.4	0.9900	25
f^{total}	34.0	0.9914	8
f^{segment}	34.0	0.9258	109
$f^{\text{segment}} + f^{\text{pause}}$	33.9	0.9294	176
Source durations + f^{segment}	34.1	0.6214	103
$f^{\text{total}} + f^{\text{segment}}$	33.9	0.9191	20

Table 6.3: Counter ablation results on the test set. We remove the auxiliary counters and/or source segment durations and measure the effect. *#Pause Mismatch* represents the number of sentences in the test set for which the wrong number of pauses is generated.

Counter Ablations

To evaluate the effectiveness of the counters and source duration tags, we start with the highest-quality factored model – embedding sizes 64, 64, 64, 32 with zeroed counter loss weights – and measure the change in translation accuracy and speech overlap for models with one or more of the counters removed. From Table 6.3, we can see that removing either the source segment tags or f^{total} has very little impact on the speech overlap, since the model is able to track the timing information from f^{segment}. Removing both the source segment tags and f^{segment} causes a large drop in speech overlap since the model then has no information about segment durations. We also see that removing f^{segment} and f^{pause} causes a large number of outputs to have the wrong number of pauses.[11] These results are consistent with our intuition about the purpose of each of these counters.

Lip-sync

Lip synchronisation, or lip-sync, is an important feature of dubbing. It is important that the final generated audio is in sync with the lip movements of the on-screen speaker in the original video. As an analysis, we looked at Lip-Sync Error – Distance (LSE-D) (Chung and Zisserman, 2017) following the evaluation methodology in Hu et al. (2021). LSE-D is not a perfect metric but it is an indication of the amount of lip-sync errors in the video. As we see in Table 6.4, Subset 1 (a random subset of videos without pauses) consistently has a lower lip-sync error than Subset 2 (a sample of longer videos

[11] We cannot remove only f^{pause} since f^{segment} uses f^{pause} to fetch the correct segment durations.

Model	Human MOS ↑	LSE-D ↓	
		Subset 1	Subset 2
Original (source)		7.39	7.67
Text to phonemes	3.16 ± 0.19	11.64	13.31
Interleaved	3.33 ± 0.18	11.71	12.35
Factored	3.43 ± 0.19	11.73	12.48

Table 6.4: Human evaluation results and lip-sync scores. Subset 1 is a random sample of 91 videos without pauses, while Subset 2 is a sample of 101 videos from the longest 10% of videos, all containing one or more pauses.

containing pauses) in all cases, showing that it is more difficult to generate lip-synced dubs for Subset 2. This result is also in line with human judgements we obtained for the two subsets where the annotators preferred dubs for Subset 1 (see Section 6.7). Secondly, original videos show significantly lower LSE-D (12.x vs 7.x) than dubbed videos, showing that automatic dubbing research still has a long way to go to reach lip-sync quality in original videos.

6.6 Qualitative Perception

Our initial intention was to perform human evaluation of dubbed videos using crowd source workers but a pilot showed very noisy results, with annotators often appearing to ignore annotation guidelines. We believe this is due at least in part to the large number of factors that affect perception of a dubbed video, including (but not limited to) translation quality, speech quality/naturalness, isochrony, and lip sync. In this section, we present instead some qualitative conclusions drawn after watching/listening to many samples.

- The baseline tends to be the most natural sounding, but the lack of isochrony is disconcerting. It is probable that the lack of isochrony would be even more jarring when viewing dubbed content with multiple speakers.
- The proposed models with little or no noise added have much better isochrony, as expected, but often sound a little more robotic than the baseline, and it is not unusual to have a word at the end of a segment repeated – presumably this happens when the translation model finishes a translation but the counters tell the model it should keep producing output.

- The proposed models with large amounts of noise also sound a bit unnatural, but for a very different reason. The speech in the test set appears to be fairly slow compared to the training data, while the model produces speech with speaking rates similar to the training data, resulting in speech segments which are often short, resulting in long, often unnatural pauses between speech segments.
- The proposed models with noise of around 0.1 seem to be the best compromise between isochrony and naturalness/translation quality, consistent with the automatic evaluation (see Figure 6.3).

6.7 Human Evaluation

The IWSLT 2023 shared task on automatic dubbing[12] (Agarwal et al., 2023) evaluated the quality of automatic dubbing systems through human evaluation. Since the models from this chapter were used as a baseline for the shared task, we obtained some human judgements of our dubbed outputs, which are presented here.

The dubbed English videos produced by the different models were judged by the authors of Pal et al. (2023), all of whom were researchers in automatic dubbing, and included both native and non-native English speakers. The judges were shown system outputs for each video in the test set in random order and asked to rate them from 1 to 6. There was no defined rubric or evaluation guidelines to follow but there was an effort to be consistent across examples and systems. We report Mean Opinion Score (MOS) in Table 6.4, where the scores for a system as judged by humans are averaged in one score.

We also looked at MOS for two different subsets of the data – Subset 1, a random sample of 91 videos with no pauses in the speech, and Subset 2, consisting of 101 videos sampled from the longest 10% of videos, where there are always one or more pauses – to analyse the relative difficulty of dubbing these different kinds of videos. We find that Subset 1 has a higher MOS of 3.54 (± 0.11) compared to Subset 2 with a MOS of 3.31 (± 0.11). This shows that it is significantly more difficult for all systems to dub Subset 2 than Subset 1, i.e. it is harder to maintain isochrony for longer videos with pauses.

[12] https://iwslt.org/2023/dubbing

6.8 Conclusions

In this chapter, we have shown that we can train models with target factors for duration prediction as well as other auxiliary counters to further guide the model and provide timing information to the model to enable isochronous translation. We have shown that target factors can be used to predict phoneme durations alongside translated phoneme sequences to jointly optimise translation and timing for automatic dubbing. Automatic evaluation shows that our models out-perform a baseline of training a model to generate interleaved phoneme and duration sequences. Although we find that all evaluated systems produce lower-scoring translations based on automatic quality metrics compared to the non-isochronous baseline, and have worse lip-sync than the original videos, human evaluation shows that our model outputs are perceived more favourably by human judges than baselines. Adding the extra timing constraints and enabling the MT model to incorporate this additional information thus results in an overall improved user experience of automatically dubbed video.

Chapter 7

Conclusions and Future Work

In this thesis, we have investigated the limitations of sentence-level MT and explored some ways in which they can be improved by enriching them with additional information.

The first part of the thesis focused on analysis of information that is missing from sentence-level MT models. In Chapter 3, we introduced the “cheat codes” method as an analysis tool. By allowing the model access to a controllable amount of information from the target, we were able to estimate the amount of information that a neural MT model needs in addition to the source sentence to reproduce reference translations. This showed us that only 2 floats worth of information per target token was enough to achieve near-perfect translation. This indicated that models augmented with only a small amount of information alongside source sentences could potentially achieve better translation quality, and that this is unlikely to be a limitation of model capacity, but instead of the data provided to the model.

In Chapter 4, we turned to the fact that even though the cheat code models were able to achieve almost perfect translations, they were unable to exactly reproduce every reference translation, even with a large amount of added target-side information to help them. Taking this as an indication that the models were unable to learn certain aspects of the languages or the translation task itself, we analysed the outputs of these models to identify the types of errors they made. We identified several weaknesses of these models, including specific parts of speech and categories of named entities that are more commonly mistranslated. We also found that multi-word named entities are often mistranslated, even when seen frequently in training data. These findings signpost a set of hard problems for neural MT for future research to focus on.

In the second part of the thesis, we moved on to specific cases of sentence-level MT models being enriched with other information. Chapter 5 looked at context-aware MT, where the model captures information from document context. We described the creation of large-scale datasets for document-level MT in several language pairs, which has typically been a bottleneck for research in this area. We then built models that use context information to improve translation quality. We performed detailed evaluations and analyses showing that the models are also able to improve their accuracy on several types of discourse phenomena that are difficult or impossible for sentence-level models, such as anaphora resolution and pronoun translation. This confirms that if we have access to enough parallel data with context, we can build models that are better and more similar to human translators who would always use context to guide their translation.

The second and final example of incorporating additional information in MT models that we explored was the use of timing information for isochronous MT. In Chapter 6, we showed that we can use target factors and auxiliary counters to provide the model with information about the durations of speech segments and keep track of the timing. By doing so, we can produce translations that are better suited for automatic dubbing and result in dubbed videos that are more favourably perceived by human evaluators. This shows a scenario where the sentence-level model can be augmented to obey external constraints and produce translations that are more useful for a specific application.

Through this work, we emphasise the fact that purely sentence-level MT is a severely limited formulation of the task, and it only became the standard formulation of MT because of the lack of large-scale datasets with context or other types of information, and due to computation tractability concerns with older models and hardware. We emphasise the fact that translation cannot be done one isolated sentence at a time, and is instead a function of many other factors. We have shown that we can enrich these models with many kinds of additional information to build MT systems that are more accurate and more useful across a wide range of applications, and hope that research in the field moves beyond the sentence-level paradigm rapidly.

7.1 Future Work

The work in this thesis identifies and opens up several directions for future research. We outline some interesting and promising directions below.

Conditional Cheat Codes

The cheat codes method introduced in Chapter 3 provided us a way to estimate an upper bound on the amount of information that a model would need in addition to source sentences to achieve perfect translation. However, this method is not able to vary the amount of information provided to the model based on the input sentence. It is clear that some sentences can be translated without any additional information while some sentences are difficult or impossible to translate perfectly on their own. Conditioning the amount of target-side information accessed by the decoder, i.e. the size of the cheat code, on the content of the source sentence would enable us to measure the amount of missing information from individual sentences required for translation, and thus provide a measure of difficulty for a given sentence to be translated. This would be an interesting avenue for future research.

Addressing and Evaluating Hard Problems

In Chapter 4, we identified several aspects of translation that are difficult to learn for neural MT models. Considering the high quality of modern MT in general, it is worth dedicating research efforts to granular issues such as these and attempt to overcome these limitations. Test sets to evaluate model performance on specific phenomena that have been identified as weaknesses would also help track progress with better models.

Investigating the Fleetwood Mac Problem

Particularly concerning among the findings of Chapter 4 was the mistranslation of relatively frequent named entities – that we termed the "Fleetwood Mac Problem". It is usually assumed that badly translated named entities are due to them being out-of-vocabulary or rare, but our initial analysis showed that the problem is not due to very low frequency, or subword segmentation issues. We suggest a possible solution in Chapter 4: allowing frequent named entities to be represented as a single token. Since named entity errors affect the perception of translation significantly, it should be investigated why this happens and how it can be addressed, and this could be an area for future research.

Modelling More Voice Information for Automatic Dubbing

In Chapter 6, we showed how phoneme durations can be predicted jointly with translation to produce isochronous translations. This could be extended to other aspects of speech, such as pitch, energy, etc., which is currently predicted by the TTS module only from the translated phonemes. While this is likely to have a smaller impact on the translations themselves, the joint modelling of these aspects could improve the overall quality of automatic dubbing.

More Document-Level MT

While document-level MT is a well-established field of research, it is still not as widely used as sentence-level MT. The creation of large-scale datasets like the ones we introduced in Chapter 5 and by Wicks et al. (2024) should push the field to concentrate on this paradigm and move away from sentence-level MT.

LLM Methods

LLMs have been shown to be effective for MT (see Section 2.2.4), especially for high-resource language pairs (Robinson et al., 2023), and look likely to take over as the dominant paradigm for the task. However, despite their inherent ability to model longer context, they are still used for MT at the sentence level, with only a few exceptions (Karpinska and Iyyer, 2023; Petrick et al., 2023; Wang et al., 2023; Wu et al., 2024; Zhang et al., 2023). Sentence-level MT using LLMs is an unnecessary constraint that should be removed. The ability of LLMs to model context should be exploited further to build better MT systems. Given their capacity to capture various kinds of information, LLMs could also be used to model many other kinds of additional information that we have discussed in this thesis.

As the field of NLP moves towards larger models, multimodal input, and huge context windows (Bulatov et al., 2024; Chen et al., 2023; Xiong et al., 2024), it is worth thinking about how much information is actually necessary to perform the real-world tasks we are interested in, and find ways to provide this information to the models in targeted ways instead of arbitrarily limiting them to sentence-level inputs or computing massive but potentially irrelevant contexts.

Appendix A

Samples from Document-Level Parallel Corpora

In this appendix, we present some examples from the datasets presented in Chapter 5 to demonstrate the format and content of the extracted data. The contexts have been truncated here due to space constraints, but they still illustrate the usefulness of the context to the translation of the sentence. Note that while these examples have been hand-picked and generally show useful context, we still see some noise in the contexts like text fragments in the wrong language or the presence of many short uninformative lines.

eng→fra with target context

`Source sentence:` Its entwined cobbled pathways and network of tunnels underground are interesting to look round.

`Target context:` EXCURSIONS À PARTIR DE PRAGUE <docline> | Croisière sur la rivière Vltava avec dîner | Shuttle d'aéroport | <docline> . . . . . . <docline> Il s'agit d'une mine d'or exposée en Bohème, avec des châteaux magiques et des petites villes éparpillées dans la campagne, et touchée par les forêts denses de la chaîne de montagne Šumava le long de la frontière avec l'Autriche. La petite ville de Tábor est charmante, et elle a été habitée par les taborites au quinzième siècle.

`Target sentence:` Ses chemins pavés entrelacés et son réseau de tunnels souterrains sont intéressants à observer.

pol→eng with target context

In this example, we see some noise in the form of Polish text fragments appearing in the English context.

`Source sentence:` Stypendium można przeznaczyć na dowolny cel.
`Target context:` START 2020 recruitment launched - Fundacja na rzecz Nauki PolskiejFundacja na rzecz Nauki Polskiej <docline> ...... <docline> The principal criteria in the competition are the quality and originality of the candidate's scientific accomplishments to date, as well as his or her single most important research achievement. In recent years the amount of the one-year stipend has been PLN 28,000.
`Target sentence:` The stipend may be used by the laureate for any purpose.

eng→deu with source context
`Source sentence:` The evening will bring clouds with rain or sleet.
`Source context:` °C <docline> 2 °C <docline> 2 °C <docline> 2 °C <docline> 2 °C <docline> 1 °C <docline> Air pressure <docline> 1014 hPa <docline> ...... <docline> Tomorrow <docline> In the early morning it will be mainly cloudy, but mostly dry. Before noon clouds with rain or sleet will dominate.
`Target sentence:` Die Mittagszeit bringt wechselhaftes Wetter mit ab und zu etwas Regen.

eng→fra with both contexts
This example contains broken sentence fragments, but is a good example of the context being required to disambiguate a pronoun in the target French – the translation of "it" to "la" requires context about the grammatical gender of the antecedent.
`Source sentence:` Previously, it appeared only below 1,600 metres.
`Source context:` aside to give to my sister." <docline> The family used coffee revenues to pay his sister's schooling, and his brother's. Mr. Zikusoka followed in his father's footsteps so that he too could provide for his family. When he got married in 2005, he received a half-hectare of farmland and decided to grow coffee. <docline> ...... <docline> He notes that coffee leaf rust disease, which usually affects coffee at altitudes lower than 1,400 metres, has now surfaced at 1,800 metres above sea level. <docline> Coffee berry disease has also shifted to higher altitudes, and is attacking crops at 1,800 metres above sea level.
`Target context:` sa production de café n'a cessé de baisser en raison de maladies et d'insectes nuisibles. Une maladie fongique appelée flétrissement du café a envahi son exploitation, et les perceurs de la tige de café ont attaqué ses caféiers. De nombreux autres agriculteurs ont subi la crise du flétrissement dans le district de Mukono, l'une des principales régions productrices de café en Ouganda. <docline> ...... <docline>

Il note que la rouille orangée du caféier qui touchait généralement le café cultivé à des altitudes inférieures à 1 400 mètres est maintenant apparue dans les localités situées à 1 800 mètres au-dessus de la mer. <docline> La maladie du fruit du caféier s'est également déportée vers les altitudes plus hautes, et est en train d'attaquer les cultures situées à 1 800 mètres au-dessus du niveau de la mer.

`Target sentence:` Autrefois, on ne la trouvait qu'aux endroits situés en dessous de 1 600 mètres.

Bibliography

Agarwal, M., Agrawal, S., Anastasopoulos, A., Bentivogli, L., Bojar, O., Borg, C., Carpuat, M., Cattoni, R., Cettolo, M., Chen, M., Chen, W., Choukri, K., Chronopoulou, A., Currey, A., Declerck, T., Dong, Q., Duh, K., Estève, Y., Federico, M., Gahbiche, S., Haddow, B., Hsu, B., Mon Htut, P., Inaguma, H., Javorský, D., Judge, J., Kano, Y., Ko, T., Kumar, R., Li, P., Ma, X., Mathur, P., Matusov, E., McNamee, P., P. McCrae, J., Murray, K., Nadejde, M., Nakamura, S., Negri, M., Nguyen, H., Niehues, J., Niu, X., Kr. Ojha, A., E. Ortega, J., Pal, P., Pino, J., van der Plas, L., Polák, P., Rippeth, E., Salesky, E., Shi, J., Sperber, M., Stüker, S., Sudoh, K., Tang, Y., Thompson, B., Tran, K., Turchi, M., Waibel, A., Wang, M., Watanabe, S., and Zevallos, R. (2023). FINDINGS OF THE IWSLT 2023 EVALUATION CAMPAIGN. In Salesky, E., Federico, M., and Carpuat, M., editors, *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 1–61, Toronto, Canada (in-person and online). Association for Computational Linguistics.

Aharoni, R., Johnson, M., and Firat, O. (2019). Massively multilingual neural machine translation. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Ahmad, I. S., Anastasopoulos, A., Bojar, O., Borg, C., Carpuat, M., Cattoni, R., Cettolo, M., Chen, W., Dong, Q., Federico, M., Haddow, B., Javorský, D., Krubiński, M., Kim Lam, T., Ma, X., Mathur, P., Matusov, E., Maurya, C., McCrae, J., Murray, K., Nakamura, S., Negri, M., Niehues, J., Niu, X., Ojha, A. K., Ortega, J., Papi, S., Polák, P., Pospíšil, A., Pecina, P., Salesky, E., Sethiya, N., Sarkar, B., Shi, J., Sikasote, C., Sperber, M., Stüker, S., Sudoh, K., Thompson, B., Waibel, A., Watanabe, S., Wilken, P., Zemánek, P., and Zevallos, R. (2024). FINDINGS OF THE IWSLT 2024 EVALUATION CAMPAIGN. In Salesky, E., Federico, M., and Carpuat, M., editors, *Proceedings of the 21st International Conference on Spoken Language Translation (IWSLT 2024)*, pages 1–11, Bangkok, Thailand (in-person and online). Association for Computational Linguistics.

Akhbardeh, F., Arkhangorodsky, A., Biesialska, M., Bojar, O., Chatterjee, R., Chaudhary, V., Costa-jussa, M. R., España-Bonet, C., Fan, A., Federmann, C., Freitag, M., Graham, Y., Grundkiewicz, R., Haddow, B., Harter, L., Heafield, K., Homan, C., Huck, M., Amponsah-Kaakyire, K., Kasai, J., Khashabi, D., Knight, K., Kocmi, T., Koehn, P., Lourie, N., Monz, C., Morishita, M., Nagata, M., Nagesh, A., Nakazawa, T., Negri, M., Pal, S., Tapo, A. A., Turchi, M., Vydrin, V., and

Zampieri, M. (2021). Findings of the 2021 conference on machine translation (WMT21). In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussa, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Kocmi, T., Martins, A., Morishita, M., and Monz, C., editors, *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online. Association for Computational Linguistics.

Al Ghussin, Y., Zhang, J., and van Genabith, J. (2023). Exploring paracrawl for document-level neural machine translation. In Vlachos, A. and Augenstein, I., editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1304–1310, Dubrovnik, Croatia. Association for Computational Linguistics.

Al-Onaizan, Y. and Knight, K. (2002). Translating named entities using monolingual and bilingual resources. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 400–408, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Artetxe, M. and Schwenk, H. (2019). Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Arthur, P., Neubig, G., and Nakamura, S. (2016). Incorporating discrete translation lexicons into neural machine translation. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, Austin, Texas. Association for Computational Linguistics.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.*

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Goldstein, J., Lavie, A., Lin, C.-Y., and Voss, C., editors, *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Bañón, M., Chen, P., Haddow, B., Heafield, K., Hoang, H., Esplà-Gomis, M., Forcada, M. L., Kamran, A., Kirefu, F., Koehn, P., Ortiz Rojas, S., Pla Sempere, L., Ramírez-Sánchez, G., Sarrías, E., Strelec, M., Thompson, B., Waites, W., Wiggins, D., and Zaragoza, J. (2020). ParaCrawl: Web-scale acquisition of parallel corpora. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Bar-Hillel, Y. (1960). The present status of automatic translation of languages. In *Advances in computers*, volume 1, pages 91–163.

Barrault, L., Biesialska, M., Bojar, O., Costa-jussà, M. R., Federmann, C., Graham, Y., Grundkiewicz, R., Haddow, B., Huck, M., Joanis, E., Kocmi, T., Koehn, P., Lo, C.-k., Ljubešić, N., Monz, C., Morishita, M., Nagata, M., Nakazawa, T., Pal, S., Post, M., and Zampieri, M. (2020). Findings of the 2020 conference on machine translation (WMT20). In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Graham, Y., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., and Negri, M., editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Bawden, R., Sennrich, R., Birch, A., and Haddow, B. (2018). Evaluating discourse phenomena in neural machine translation. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Bergmanis, T. and Pinnis, M. (2021). Facilitating terminology translation with target lemma annotations. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3105–3111, Online. Association for Computational Linguistics.

Bojar, O. (2007). English-to-czech factored machine translation. In *Proceedings of the second workshop on statistical machine translation*, pages 232–239.

Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Huck, M., Hokamp, C., Koehn, P., Logacheva, V., Monz, C., Negri, M., Post, M., Scarton, C., Specia, L., and Turchi, M. (2015). Findings of the 2015 workshop on statistical machine translation. In Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Hokamp, C., Huck, M., Logacheva, V., and Pecina, P., editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal. Association for Computational Linguistics.

Bojar, O., Dušek, O., Kocmi, T., Libovický, J., Novák, M., Popel, M., Sudarikov, R., and Variš, D. (2016). Czeng 1.6: Enlarged czech-english parallel corpus with processing tools dockered. In Sojka, P., Horák, A., Kopeček, I., and Pala, K., editors, *Text, Speech, and Dialogue*, pages 231–238, Cham. Springer International Publishing.

Bouthors, M., Crego, J., and Yvon, F. (2024). Retrieving examples from memory for retrieval augmented neural machine translation: A systematic comparison. In Duh, K., Gomez, H., and Bethard, S., editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3022–3039, Mexico City, Mexico. Association for Computational Linguistics.

Brannon, W., Virkar, Y., and Thompson, B. (2023). Dubbing in practice: A large scale study of human localization with insights for automatic dubbing. *Transactions of the Association for Computational Linguistics*, 11:419–435.

Brown, R. and Gilman, A. (1968). *THE PRONOUNS OF POWER AND SOLIDARITY*, pages 252–275. De Gruyter Mouton, Berlin, Boston.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Bugliarello, E., Mielke, S. J., Anastasopoulos, A., Cotterell, R., and Okazaki, N. (2020). It's easier to translate out of English than into it: Measuring neural translation difficulty by cross-mutual information. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1640–1649, Online. Association for Computational Linguistics.

Bulatov, A., Kuratov, Y., Kapushev, Y., and Burtsev, M. S. (2024). Scaling transformer to 1m tokens and beyond with rmt.

Bulte, B. and Tezcan, A. (2019). Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1800–1809, Florence, Italy. Association for Computational Linguistics.

Burchardt, A. (2013). Multidimensional quality metrics: a flexible system for assessing translation quality. In *Proceedings of Translating and the Computer 35*, London, UK. Aslib.

Caglayan, O., Barrault, L., and Bougares, F. (2016). Multimodal attention for neural machine translation.

Cai, D., Wang, Y., Li, H., Lam, W., and Liu, L. (2021). Neural machine translation with monolingual translation memory. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7307–7318, Online. Association for Computational Linguistics.

Chen, P., Helcl, J., Germann, U., Burchell, L., Bogoychev, N., Miceli Barone, A. V., Waldendorf, J., Birch, A., and Heafield, K. (2021). The University of Edinburgh's English-German and English-Hausa submissions to the WMT21 news translation task. In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussa, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Kocmi, T., Martins, A., Morishita, M., and Monz, C., editors, *Proceedings of the Sixth Conference on Machine Translation*, pages 104–109, Online. Association for Computational Linguistics.

Chen, S., Wong, S., Chen, L., and Tian, Y. (2023). Extending context window of large language models via positional interpolation.

Cho, I., Wang, D., Takahashi, R., and Saito, H. (2022). A personalized dialogue generator with implicit user persona detection. In Calzolari, N., Huang, C.-R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.-S., Ryu, P.-M., Chen, H.-H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S.-H., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 367–377, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation: Encoder–decoder approaches. In Wu, D., Carpuat, M., Carreras, X., and Vecchi, E. M., editors, *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Chronopoulou, A., Thompson, B., Mathur, P., Virkar, Y., Lakew, S. M., and Federico, M. (2023). Jointly optimizing translations and speech timing to improve isochrony in automatic dubbing.

Chu, C., Dabre, R., and Kurohashi, S. (2017). An empirical comparison of domain adaptation methods for neural machine translation. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–391, Vancouver, Canada. Association for Computational Linguistics.

Chung, J. S. and Zisserman, A. (2017). Out of time: Automated lip sync in the wild. In Chen, C.-S., Lu, J., and Ma, K.-K., editors, *Computer Vision – ACCV 2016 Workshops*, pages 251–263, Cham. Springer International Publishing.

Dinu, G., Mathur, P., Federico, M., and Al-Onaizan, Y. (2019). Training neural machine translation to apply terminology constraints. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Duquenne, P.-A., Gong, H., Sagot, B., and Schwenk, H. (2022). T-modules: Translation modules for zero-shot cross-modal machine translation. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5794–5806, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Edunov, S., Ott, M., Auli, M., and Grangier, D. (2018). Understanding back-translation at scale. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

El-Kishky, A., Chaudhary, V., Guzmán, F., and Koehn, P. (2020). CCAligned: A massive collection of cross-lingual web-document pairs. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5960–5969, Online. Association for Computational Linguistics.

España-Bonet, C. and van Genabith, J. (2018). Multilingual semantic networks for data-driven interlingua seq2seq systems. In Du, J., Arcan, M., Liu, Q., and Isahara, H., editors, *Proceedings of the LREC 2018 Workshop "MLP-MomenT". International Conference on Language Resources and Evaluation (LREC-2018), located at 20th, May 7-12, Miyazaki, Japan*, pages 8–13.

Federico, M., Enyedi, R., Barra-Chicote, R., Giri, R., Isik, U., Krishnaswamy, A., and Sawaf, H. (2020a). From speech-to-speech translation to automatic dubbing. In Federico, M., Waibel, A., Knight, K., Nakamura, S., Ney, H., Niehues, J., Stüker, S., Wu, D., Mariani, J., and Yvon, F., editors, *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 257–264, Online. Association for Computational Linguistics.

Federico, M., Virkar, Y., Enyedi, R., and Barra-Chicote, R. (2020b). Evaluating and Optimizing Prosodic Alignment for Automatic Dubbing. In *Proc. Interspeech 2020*, pages 1481–1485.

Feng, F., Yang, Y., Cer, D., Arivazhagan, N., and Wang, W. (2022). Language-agnostic BERT sentence embedding. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Feng, Y., Zhang, S., Zhang, A., Wang, D., and Abel, A. (2017). Memory-augmented neural machine translation. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1390–1399, Copenhagen, Denmark. Association for Computational Linguistics.

Fernandes, P., Yin, K., Neubig, G., and Martins, A. F. T. (2021). Measuring and increasing context usage in context-aware machine translation. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6467–6478, Online. Association for Computational Linguistics.

Fishel, M., Bojar, O., Zeman, D., and Berka, J. (2011). Automatic translation error analysis. In Habernal, I. and Matoušek, V., editors, *Text, Speech and Dialogue*, pages 72–79, Berlin, Heidelberg. Springer Berlin Heidelberg.

Gao, Y., Wang, R., and Hou, F. (2023). How to design translation prompts for chatgpt: An empirical study.

Garcia, X., Bansal, Y., Cherry, C., Foster, G., Krikun, M., Johnson, M., and Firat, O. (2023). The unreasonable effectiveness of few-shot learning for machine translation. In *International Conference on Machine Learning*, pages 10867–10878. PMLR.

García-Martínez, M., Barrault, L., and Bougares, F. (2016). Factored neural machine translation architectures. In Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., Cattoni, R., and Federico, M., editors, *Proceedings of the 13th International Conference on Spoken Language Translation*, Seattle, Washington D.C. International Workshop on Spoken Language Translation.

Guan, J., Mao, X., Fan, C., Liu, Z., Ding, W., and Huang, M. (2021). Long text generation by modeling sentence-level and discourse-level coherence. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6379–6393, Online. Association for Computational Linguistics.

Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., and Bengio, Y. (2016). Pointing the unknown words. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 140–149, Berlin, Germany. Association for Computational Linguistics.

Hasler, E., de Gispert, A., Iglesias, G., and Byrne, B. (2018). Neural machine translation decoding with terminology constraints. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 506–512, New Orleans, Louisiana. Association for Computational Linguistics.

Hassan, H., Aue, A., Chen, C., Chowdhary, V., Clark, J., Federmann, C., Huang, X., Junczys-Dowmunt, M., Lewis, W., Li, M., Liu, S., Liu, T.-Y., Luo, R., Menezes, A., Qin, T., Seide, F., Tan, X., Tian, F., Wu, L., Wu, S., Xia, Y., Zhang, D., Zhang, Z., and Zhou, M. (2018). Achieving human parity on automatic chinese to english news translation.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

He, Q., Huang, G., Cui, Q., Li, L., and Liu, L. (2021). Fast and accurate neural machine translation with translation memory. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3170–3180, Online. Association for Computational Linguistics.

He, X., Haffari, G., and Norouzi, M. (2018). Sequence to sequence mixture model for diverse machine translation. In Korhonen, A. and Titov, I., editors, *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 583–592, Brussels, Belgium. Association for Computational Linguistics.

Hendy, A., Abdelrehim, M., Sharaf, A., Raunak, V., Gabr, M., Matsushita, H., Kim, Y. J., Afify, M., and Awadalla, H. H. (2023). How good are gpt models at machine translation? a comprehensive evaluation.

Hoang, C., Sachan, D., Mathur, P., Thompson, B., and Federico, M. (2023). Improving retrieval augmented neural machine translation by controlling source and fuzzy-match interactions. In Vlachos, A. and Augenstein, I., editors, *Findings of the Association for Computational Linguistics: EACL 2023*, pages 289–295, Dubrovnik, Croatia. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hu, C., Tian, Q., Li, T., Yuping, W., Wang, Y., and Zhao, H. (2021). Neural dubber: Dubbing for videos according to scripts. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 16582–16595. Curran Associates, Inc.

Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. (2017). Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898.

Isabelle, P., Cherry, C., and Foster, G. (2017). A challenge set approach to evaluating machine translation. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2486–2496, Copenhagen, Denmark. Association for Computational Linguistics.

Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2015). On using very large target vocabulary for neural machine translation. In Zong, C. and Strube, M., editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China. Association for Computational Linguistics.

Jean, S., Lauly, S., Firat, O., and Cho, K. (2017). Does neural machine translation benefit from larger context? *arXiv preprint arXiv:1704.05135*.

Jiao, W., Wang, W., tse Huang, J., Wang, X., Shi, S., and Tu, Z. (2023). Is chatgpt a good translator? yes with gpt-4 as the engine.

Johnson, M. (2018). Providing gender-specific translations in Google Translate. https://research.google/blog/providing-gender-specific-translations-in-google-translate/. [Online; posted 10-December-2018; accessed 29-May-2024].

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2017). Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Jon, J. (2019). discourse-test-set. https://github.com/cepin19/discourse-test-set. [Online; accessed 22-May-2024].

Junczys-Dowmunt, M. (2019). Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Turchi, M., and Verspoor, K., editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 225–233, Florence, Italy. Association for Computational Linguistics.

Junczys-Dowmunt, M. and Grundkiewicz, R. (2017). An exploration of neural sequence-to-sequence architectures for automatic post-editing. In Kondrak, G. and Watanabe, T., editors, *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 120–129, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Junczys-Dowmunt, M. and Grundkiewicz, R. (2018). MS-UEdin submission to the WMT2018 APE shared task: Dual-source transformer for automatic post-editing. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 822–826, Belgium, Brussels. Association for Computational Linguistics.

Junczys-Dowmunt, M., Heafield, K., Hoang, H., Grundkiewicz, R., and Aue, A. (2018). Marian: Cost-effective high-quality neural machine translation in C++. In Birch, A., Finch, A., Luong, T., Neubig, G., and Oda, Y., editors, *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 129–135, Melbourne, Australia. Association for Computational Linguistics.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., and Bethard, S., editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Karpinska, M. and Iyyer, M. (2023). Large language models effectively leverage document-level context for literary translation, but critical errors persist. In Koehn, P., Haddow, B., Kocmi, T., and Monz, C., editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 419–451, Singapore. Association for Computational Linguistics.

Khandelwal, U., Fan, A., Jurafsky, D., Zettlemoyer, L., and Lewis, M. (2021). Nearest neighbor machine translation. In *International Conference on Learning Representations*.

King, M. and Falkedal, K. (1990). Using test suites in evaluation of machine translation systems. In *COLING 1990 Volume 2: Papers presented to the 13th International Conference on Computational Linguistics*.

Kobus, C., Crego, J., and Senellart, J. (2017). Domain control for neural machine translation. In Mitkov, R. and Angelova, G., editors, *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 372–378, Varna, Bulgaria. INCOMA Ltd.

Kocmi, T., Avramidis, E., Bawden, R., Bojar, O., Dvorkovich, A., Federmann, C., Fishel, M., Freitag, M., Gowda, T., Grundkiewicz, R., Haddow, B., Koehn, P., Marie, B., Monz, C., Morishita, M., Murray, K., Nagata, M., Nakazawa, T., Popel, M., Popović, M., and Shmatova, M. (2023). Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In Koehn, P., Haddow, B., Kocmi, T., and Monz, C., editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 1–42, Singapore. Association for Computational Linguistics.

Kocmi, T., Bawden, R., Bojar, O., Dvorkovich, A., Federmann, C., Fishel, M., Gowda, T., Graham, Y., Grundkiewicz, R., Haddow, B., Knowles, R., Koehn, P., Monz, C., Morishita, M., Nagata, M., Nakazawa, T., Novák, M., Popel, M., and Popović, M. (2022). Findings of the 2022 conference on machine translation (WMT22). In Koehn, P., Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Jimeno Yepes, A., Kocmi, T., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., Negri, M., Névéol, A., Neves, M., Popel, M., Turchi, M., and Zampieri, M., editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 1–45, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Kocmi, T., Popel, M., and Bojar, O. (2020). Announcing czeng 2.0 parallel corpus with over 2 gigawords. *arXiv preprint arXiv:2007.03006*.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Koehn, P. and Hoang, H. (2007). Factored translation models. In Eisner, J., editor, *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In Ananiadou, S., editor, *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In Luong, T., Birch, A., Neubig, G., and Finch, A., editors, *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Kuang, S., Xiong, D., Luo, W., and Zhou, G. (2018). Modeling coherence for neural machine translation with dynamic and topic caches. In Bender, E. M., Derczynski, L., and Isabelle, P., editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pages 596–606, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Blanco, E. and Lu, W., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Lakew, S. M., Federico, M., Wang, Y., Hoang, C., Virkar, Y., Barra-Chicote, R., and Enyedi, R. (2021). Machine translation verbosity control for automatic dubbing. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7538–7542. IEEE.

Lakew, S. M., Virkar, Y., Mathur, P., and Federico, M. (2022). Isometric MT: Neural machine translation for automatic dubbing. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6242–6246. IEEE.

Lala, C. and Specia, L. (2018). Multimodal lexical translation. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Läubli, S., Sennrich, R., and Volk, M. (2018). Has machine translation achieved human parity? a case for document-level evaluation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4796, Brussels, Belgium. Association for Computational Linguistics.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Li, B., Liu, H., Wang, Z., Jiang, Y., Xiao, T., Zhu, J., Liu, T., and Li, C. (2020). Does multi-encoder help? a case study on context-aware neural machine translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3512–3518, Online. Association for Computational Linguistics.

Li, Z., Wang, X., Aw, A. T., Chng, E. S., and Li, H. (2018). Named-entity tagging and domain adaptation for better customized translation. In Chen, N., Banchs, R. E., Duan, X., Zhang, M., and Li, H., editors, *Proceedings of the Seventh Named Entities Workshop*, pages 41–46, Melbourne, Australia. Association for Computational Linguistics.

Lin, X. V., Mihaylov, T., Artetxe, M., Wang, T., Chen, S., Simig, D., Ott, M., Goyal, N., Bhosale, S., Du, J., Pasunuru, R., Shleifer, S., Koura, P. S., Chaudhary, V., O'Horo, B., Wang, J., Zettlemoyer, L., Kozareva, Z., Diab, M., Stoyanov, V., and Li, X. (2022). Few-shot learning with multilingual generative language models. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lison, P. and Tiedemann, J. (2016). OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692.*

Lopes, A., Farajian, M. A., Bawden, R., Zhang, M., and Martins, A. F. T. (2020). Document-level neural MT: A systematic comparison. In Martins, A., Moniz, H., Fumega, S., Martins, B., Batista, F., Coheur, L., Parra, C., Trancoso, I., Turchi, M., Bisazza, A., Moorkens, J., Guerberof, A., Nurminen, M., Marg, L., and Forcada, M. L., editors, *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 225–234, Lisboa, Portugal. European Association for Machine Translation.

Luong, T., Pham, H., and Manning, C. D. (2015a). Effective approaches to attention-based neural machine translation. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Luong, T., Sutskever, I., Le, Q., Vinyals, O., and Zaremba, W. (2015b). Addressing the rare word problem in neural machine translation. In Zong, C. and Strube, M., editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 11–19, Beijing, China. Association for Computational Linguistics.

Maruf, S. and Haffari, G. (2018). Document context neural machine translation with memory networks. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1275–1284, Melbourne, Australia. Association for Computational Linguistics.

Maruf, S., Saleh, F., and Haffari, G. (2021). A survey on document-level neural machine translation: Methods and evaluation. *ACM Comput. Surv.*, 54(2).

McAuliffe, M., Socolof, M., Mihuc, S., Wagner, M., and Sonderegger, M. (2017). Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502.

Miculicich, L., Ram, D., Pappas, N., and Henderson, J. (2018). Document-level neural machine translation with hierarchical attention networks. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954, Brussels, Belgium. Association for Computational Linguistics.

Minh-Cong, N.-H., Ngo, V. T., and Nguyen, V. V. (2022). A simple and fast strategy for handling rare words in neural machine translation. In Hanqi, Y., Zonghan, Y., Ruder, S., and Xiaojun, W., editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 40–46, Online. Association for Computational Linguistics.

Miyashita, D., Lee, E. H., and Murmann, B. (2016). Convolutional neural networks using logarithmic data representation. *CoRR*, abs/1603.01025.

Mohammed, W. and Niculae, V. (2024). On measuring context utilization in document-level mt systems.

Moslem, Y., Haque, R., Kelleher, J. D., and Way, A. (2023). Adaptive machine translation with large language models. In Nurminen, M., Brenner, J., Koponen, M., Latomaa, S., Mikhailov, M., Schierl, F., Ranasinghe, T., Vanmassenhove, E., Vidal, S. A., Aranberri, N., Nunziatini, M., Escartín, C. P., Forcada, M., Popovic, M., Scarton, C., and Moniz, H., editors, *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 227–237, Tampere, Finland. European Association for Machine Translation.

Müller, M., Rios, A., Voita, E., and Sennrich, R. (2018). A large-scale test set for the evaluation of context-aware pronoun translation in neural machine translation. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 61–72, Brussels, Belgium. Association for Computational Linguistics.

Nadejde, M., Currey, A., Hsu, B., Niu, X., Federico, M., and Dinu, G. (2022). CoCoA-MT: A dataset and benchmark for contrastive controlled MT with application to formality. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 616–632, Seattle, United States. Association for Computational Linguistics.

Nădejde, M., Reddy, S., Sennrich, R., Dwojak, T., Junczys-Dowmunt, M., Koehn, P., and Birch, A. (2017). Predicting target language CCG supertags improves neural machine translation. In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., and Kreutzer, J., editors, *Proceedings of the Second Conference on Machine Translation*, pages 68–79, Copenhagen, Denmark. Association for Computational Linguistics.

Neubig, G., Dou, Z.-Y., Hu, J., Michel, P., Pruthi, D., and Wang, X. (2019). compare-mt: A tool for holistic comparison of language generation systems. In Ammar, W., Louis, A., and Mostafazadeh, N., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 35–41, Minneapolis, Minnesota. Association for Computational Linguistics.

Niu, X., Dinu, G., Mathur, P., and Currey, A. (2021). Faithful target attribute prediction in neural machine translation.

Niu, X., Martindale, M., and Carpuat, M. (2017). A study of style in machine translation: Controlling the formality of machine translation output. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2814–2819, Copenhagen, Denmark. Association for Computational Linguistics.

Pal, P., Birch, A., and Heafield, K. (2024). Document-level machine translation with large-scale public parallel corpora. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13185–13197, Bangkok, Thailand. Association for Computational Linguistics.

Pal, P. and Heafield, K. (2022). Cheat codes to quantify missing source information in neural machine translation. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2472–2477, Seattle, United States. Association for Computational Linguistics.

Pal, P. and Heafield, K. (2023). Cheating to identify hard problems for neural machine translation. In Vlachos, A. and Augenstein, I., editors, *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1620–1631, Dubrovnik, Croatia. Association for Computational Linguistics.

Pal, P., Thompson, B., Virkar, Y., Mathur, P., Chronopoulou, A., and Federico, M. (2023). Improving Isochronous Machine Translation with Target Factors and Auxiliary Counters. In *Proc. INTERSPEECH 2023*, pages 37–41.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Petrick, F., Herold, C., Petrushkov, P., Khadivi, S., and Ney, H. (2023). Document-level language models for machine translation. In Koehn, P., Haddow, B., Kocmi, T., and Monz, C., editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 375–391, Singapore. Association for Computational Linguistics.

Petrov, S., Das, D., and McDonald, R. (2012). A universal part-of-speech tagset. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2089–2096, Istanbul, Turkey. European Language Resources Association (ELRA).

Popović, M. (2015). chrF: character n-gram F-score for automatic MT evaluation. In Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Hokamp, C., Huck, M., Logacheva, V., and Pecina, P., editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Popović, M. and Ney, H. (2011). Towards automatic error analysis of machine translation output. *Computational Linguistics*, 37(4):657–688.

Post, M. (2018). A call for clarity in reporting BLEU scores. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Post, M., Gowda, T., Grundkiewicz, R., Khayrallah, H., Jain, R., and Junczys-Dowmunt, M. (2023). SOTASTREAM: A streaming approach to machine translation training. In Tan, L., Milajevs, D., Chauhan, G., Gwinnup, J., and Rippeth, E., editors, *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pages 110–119, Singapore. Association for Computational Linguistics.

Post, M. and Junczys-Dowmunt, M. (2023). Escaping the sentence-level paradigm in machine translation. *arXiv preprint arXiv:2304.12959*.

Quan, J. and Xiong, D. (2020). Modeling long context for task-oriented dialogue state generation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7119–7124, Online. Association for Computational Linguistics.

Rabinovich, E., Patel, R. N., Mirkin, S., Specia, L., and Wintner, S. (2017). Personalized machine translation: Preserving original author traits. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1074–1084, Valencia, Spain. Association for Computational Linguistics.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. (2016). Sequence level training with recurrent neural networks. In Bengio, Y. and LeCun, Y., editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Rei, R., C. de Souza, J. G., Alves, D., Zerva, C., Farinha, A. C., Glushkova, T., Lavie, A., Coheur, L., and Martins, A. F. T. (2022). COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In Koehn, P., Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Jimeno Yepes, A., Kocmi, T., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., Negri, M., Névéol, A., Neves, M., Popel, M., Turchi, M., and Zampieri, M., editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Rei, R., Farinha, A. C., Stewart, C., Coheur, L., and Lavie, A. (2021). MT-Telescope: An interactive platform for contrastive evaluation of MT systems. In Ji, H., Park, J. C., and Xia, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 73–80, Online. Association for Computational Linguistics.

Rei, R., Stewart, C., Farinha, A. C., and Lavie, A. (2020). COMET: A neural framework for MT evaluation. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Roberts, N., Liang, D., Neubig, G., and Lipton, Z. (2020). Decoding and diversity in machine translation. In *NeurIPS 2020 Workshop on Resistance AI*.

Robinson, N., Ogayo, P., Mortensen, D. R., and Neubig, G. (2023). ChatGPT MT: Competitive for high- (but not low-) resource languages. In Koehn, P., Haddow, B., Kocmi, T., and Monz, C., editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 392–418, Singapore. Association for Computational Linguistics.

S., S. T., Tandon, S., and Bauer, R. (2017). A dual encoder sequence to sequence model for open-domain dialogue modeling. *CoRR*, abs/1710.10520.

Saboo, A. and Baumann, T. (2019). Integration of dubbing constraints into machine translation. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Turchi, M., and Verspoor, K., editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 94–101, Florence, Italy. Association for Computational Linguistics.

Sellam, T., Das, D., and Parikh, A. (2020). BLEURT: Learning robust metrics for text generation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Sennrich, R. (2017). How grammatical is character-level neural machine translation? assessing MT quality with contrastive translation pairs. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 376–382, Valencia, Spain. Association for Computational Linguistics.

Sennrich, R. and Haddow, B. (2016). Linguistic input features improve neural machine translation. In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Guillou, L., Haddow, B., Huck, M., Yepes, A. J., Névéol, A., Neves, M., Pecina, P., Popel, M., Koehn, P., Monz, C., Negri, M., Post, M., Specia, L., Verspoor, K., Tiedemann, J., and Turchi, M., editors, *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2016a). Controlling politeness in neural machine translation via side constraints. In Knight, K., Nenkova, A., and Rambow, O., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2016b). Improving neural machine translation models with monolingual data. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2016c). Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Song, K., Wang, K., Yu, H., Zhang, Y., Huang, Z., Luo, W., Duan, X., and Zhang, M. (2020). Alignment-enhanced transformer for constraining nmt with pre-specified translations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8886–8893.

Stafanovičs, A., Bergmanis, T., and Pinnis, M. (2020). Mitigating gender bias in machine translation with target gender annotations. In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Graham, Y., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., and Negri, M., editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 629–638, Online. Association for Computational Linguistics.

Stergiadis, E., Kumar, S., Kovalev, F., and Levin, P. (2021). Multi-domain adaptation in neural machine translation through multidimensional tagging. In Campbell, J., Huyck, B., Larocca, S., Marciano, J., Savenkov, K., and Yanishevsky, A., editors, *Proceedings of Machine Translation Summit XVIII: Users and Providers Track*, pages 396–420, Virtual. Association for Machine Translation in the Americas.

Sun, Z., Wang, M., Zhou, H., Zhao, C., Huang, S., Chen, J., and Li, L. (2022). Rethinking document-level neural machine translation. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3537–3548, Dublin, Ireland. Association for Computational Linguistics.

Susanto, R. H., Chollampatt, S., and Tan, L. (2020). Lexically constrained neural machine translation with Levenshtein transformer. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3536–3543, Online. Association for Computational Linguistics.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Tam, D., Lakew, S. M., Virkar, Y., Mathur, P., and Federico, M. (2022). Isochrony-aware neural machine translation for automatic dubbing. In *Interspeech 2022*.

Thompson, B. and Post, M. (2020a). Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–121, Online. Association for Computational Linguistics.

Thompson, B. and Post, M. (2020b). Paraphrase generation as zero-shot multilingual translation: Disentangling semantic similarity from lexical and syntactic diversity. In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Graham, Y., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., and Negri, M., editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 561–570, Online. Association for Computational Linguistics.

Tiedemann, J. and Scherrer, Y. (2017). Neural machine translation with extended context. In Webber, B., Popescu-Belis, A., and Tiedemann, J., editors, *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark. Association for Computational Linguistics.

Toral, A., Castilho, S., Hu, K., and Way, A. (2018). Attaining the unattainable? reassessing claims of human parity in neural machine translation. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 113–123, Brussels, Belgium. Association for Computational Linguistics.

Tu, Z., Liu, Y., Shi, S., and Zhang, T. (2018). Learning to remember translation history with a continuous cache. *Transactions of the Association for Computational Linguistics*, 6:407–420.

Vanmassenhove, E., Hardmeier, C., and Way, A. (2018). Getting gender right in neural machine translation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3003–3008, Brussels, Belgium. Association for Computational Linguistics.

Vanmassenhove, E., Shterionov, D., and Way, A. (2019). Lost in translation: Loss and decay of linguistic richness in machine translation. In Forcada, M., Way, A., Haddow, B., and Sennrich, R., editors, *Proceedings of Machine Translation Summit XVII: Research Track*, pages 222–232, Dublin, Ireland. European Association for Machine Translation.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vilar, D., Freitag, M., Cherry, C., Luo, J., Ratnakar, V., and Foster, G. (2023). Prompting PaLM for translation: Assessing strategies and performance. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15406–15427, Toronto, Canada. Association for Computational Linguistics.

Vilar, D., Xu, J., D'Haro, L. F., and Ney, H. (2006). Error analysis of statistical machine translation output. In Calzolari, N., Choukri, K., Gangemi, A., Maegaard, B., Mariani, J., Odijk, J., and Tapias, D., editors, *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Virkar, Y., Federico, M., Enyedi, R., and Barra-Chicote, R. (2021). Improvements to prosodic alignment for automatic dubbing. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7543–7574.

Virkar, Y., Federico, M., Enyedi, R., and Barra-Chicote, R. (2022). Prosodic alignment for off-screen automatic dubbing. In *Proc. Interspeech 2022*, pages 496–500.

Voita, E., Sennrich, R., and Titov, I. (2019). When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Voita, E., Sennrich, R., and Titov, I. (2021). Analyzing the source and target contributions to predictions in neural machine translation. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1126–1140, Online. Association for Computational Linguistics.

Voita, E., Serdyukov, P., Sennrich, R., and Titov, I. (2018). Context-aware neural machine translation learns anaphora resolution. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Wan, Y., Yang, B., Wong, D. F., Chao, L. S., Yao, L., Zhang, H., and Chen, B. (2022). Challenges of neural machine translation for short texts. *Computational Linguistics*, 48(2):321–342.

Wang, C., Wu, A., Gu, J., and Pino, J. (2021a). Covost 2 and massively multilingual speech translation. In *Interspeech 2021*, pages 2247–2251.

Wang, L., Lyu, C., Ji, T., Zhang, Z., Yu, D., Shi, S., and Tu, Z. (2023). Document-level machine translation with large language models. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16646–16661, Singapore. Association for Computational Linguistics.

Wang, Y., Hoang, C., and Federico, M. (2021b). Towards modeling the style of translators in neural machine translation. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1193–1199, Online. Association for Computational Linguistics.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Weischedel, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., Xue, N., Taylor, A., Kaufman, J., Franchini, M., et al. (2013). Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Wicks, R. and Duh, K. (2022). The effects of language token prefixing for multilingual machine translation. In He, Y., Ji, H., Li, S., Liu, Y., and Chang, C.-H., editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 148–153, Online only. Association for Computational Linguistics.

Wicks, R., Post, M., and Koehn, P. (2024). Recovering document annotations for sentence-level bitext. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Findings of the Association for Computational Linguistics ACL 2024*, pages 9876–9890, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Wilken, P. and Matusov, E. (2019). Novel applications of factored neural machine translation.

Wilken, P. and Matusov, E. (2022). AppTek's submission to the IWSLT 2022 isometric spoken language translation task. In Salesky, E., Federico, M., and Costa-jussà, M., editors, *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 369–378, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Wu, M., Vu, T.-T., Qu, L., Foster, G., and Haffari, G. (2024). Adapting large language models for document-level machine translation.

Wu, Y., Guo, J., Tan, X., Zhang, C., Li, B., Song, R., He, L., Zhao, S., Menezes, A., and Bian, J. (2023). Videodubber: machine translation with speech-aware length control for video dubbing. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Xiong, W., Liu, J., Molybog, I., Zhang, H., Bhargava, P., Hou, R., Martin, L., Rungta, R., Sankararaman, K. A., Oguz, B., Khabsa, M., Fang, H., Mehdad, Y., Narang, S., Malik, K., Fan, A., Bhosale, S., Edunov, S., Lewis, M., Wang, S., and Ma, H. (2024). Effective long-context scaling of foundation models. In Duh, K., Gomez, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4643–4663, Mexico City, Mexico. Association for Computational Linguistics.

Xu, H., Kim, Y. J., Sharaf, A., and Awadalla, H. H. (2024a). A paradigm shift in machine translation: Boosting translation performance of large language models. In *The Twelfth International Conference on Learning Representations*.

Xu, H., Murray, K., Koehn, P., Hoang, H., Eriguchi, A., and Khayrallah, H. (2024b). X-alma: Plug & play modules and adaptive rejection for quality translation at scale.

Xu, H., Sharaf, A., Chen, Y., Tan, W., Shen, L., Durme, B. V., Murray, K., and Kim, Y. J. (2024c). Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation. In *Forty-first International Conference on Machine Learning*.

Xu, J., Crego, J., and Senellart, J. (2020). Boosting neural machine translation with similar translations. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1580–1590, Online. Association for Computational Linguistics.

Yamada, M. (2023). Optimizing machine translation through prompt engineering: An investigation into ChatGPT's customizability. In Yamada, M. and do Carmo, F., editors, *Proceedings of Machine Translation Summit XIX, Vol. 2: Users Track*, pages 195–204, Macau SAR, China. Asia-Pacific Association for Machine Translation.

Yao, K., Zhang, L., Du, D., Luo, T., Tao, L., and Wu, Y. (2020). Dual encoding for abstractive text summarization. *IEEE Transactions on Cybernetics*, 50:985–996.

Yao, S. and Wan, X. (2020). Multimodal transformer for multimodal machine translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4346–4350, Online. Association for Computational Linguistics.

Zeman, D., Fishel, M., Berka, J., and Bojar, O. (2011). Addicter: What is wrong with my translations? In *Prague Bull. Math. Linguistics*.

Zhang, B., Haddow, B., and Birch, A. (2023). Prompting large language model for machine translation: a case study. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Zhang, J., Luan, H., Sun, M., Zhai, F., Xu, J., Zhang, M., and Liu, Y. (2018). Improving the transformer translation model with document-level context. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542, Brussels, Belgium. Association for Computational Linguistics.

Zhang, T., Ye, W., Yang, B., Zhang, L., Ren, X., Liu, D., Sun, J., Zhang, S., Zhang, H., and Zhao, W. (2022). Frequency-aware contrastive learning for neural machine translation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 11712–11720.

Zhu, W., Liu, H., Dong, Q., Xu, J., Huang, S., Kong, L., Chen, J., and Li, L. (2024). Multilingual machine translation with large language models: Empirical results and analysis. In Duh, K., Gomez, H., and Bethard, S., editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2765–2781, Mexico City, Mexico. Association for Computational Linguistics.

Zoph, B. and Knight, K. (2016). Multi-source neural translation. In Knight, K., Nenkova, A., and Rambow, O., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 30–34, San Diego, California. Association for Computational Linguistics.

Öktem, A., Farrús, M., and Bonafonte, A. (2019). Prosodic Phrase Alignment for Machine Dubbing. In *Proc. Interspeech 2019*, pages 4215–4219.