Karlsruhe Institute of Technology

Semantic Segmentation for Translation Evaluation

Master's Thesis of

Daniel Appenzeller

At the KIT Department of Informatics
Institute for Anthropomatics and Robotics

First examiner:	Prof. Jan Niehues
Second examiner:	Prof. Dr.-Ing. Tamim Asfour
First advisor:	M.Sc. Tu Anh Dinh

10. July 2024 – 10. January 2025

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

Semantic Segmentation for Translation Evaluation (Master's Thesis)

I declare that I have developed and written the enclosed thesis completely by myself. I have not used any other than the aids that I have mentioned. I have marked all parts of the thesis that I have included from referenced literature, either in their original wording or paraphrasing their contents. I have followed the by-laws to implement scientific integrity at KIT.

Knittlingen, 10. January 2025

. . . . . .

(Daniel Appenzeller)

Abstract

Speech translation systems often segment longer recordings into smaller units for processing. Evaluating the quality of these translations typically involves comparing machine outputs to references created by human translators. Because these references are often segmented differently, the machine translations must be re-segmented to align with the reference segmentation. Current methods, such as mwerSegmenter, align machine outputs to references by minimizing word error rate but disregard semantic content.

This thesis introduces two semantic re-segmentation approaches aimed at addressing this limitation: one leveraging sentence embeddings for segment similarity and another using a large language model for direct re-segmentation. The goal is to improve the resulting segmentations, which improves the reliability of automatic metrics and the correlation of metric scores with human judgments. While the large language model based method demonstrates potential by avoiding common MWER issues, it is inconsistent and fails to segment many translations accurately. The semantic similarity approach performs slightly worse than MWER for segment-level correlations but often outperforms MWER at the document and system levels, especially with metrics like BERTScore. Additionally, semantic segmentation is capable of reference-free re-segmentation, enabling segment-level evaluations without a reference.

Overall, improvements over MWER segmentation yield moderate improvements in correlation for both approaches. The maximum observed improvement is approximately 0.05, with individual outliers reaching higher values for the COMET metric.

Zusammenfassung

Sprachübersetzungssysteme segmentieren längere Aufnahmen häufig in kleinere Einheiten zur Verarbeitung. Die Bewertung der Übersetzungsqualität erfolgt typischerweise durch den Vergleich von maschinellen Übersetzungen mit Referenzen, die von menschlichen Übersetzern erstellt wurden. Da diese Referenzen oft anders segmentiert sind, müssen die maschinellen Übersetzungen neu segmentiert werden, um mit der Referenz übereinzustimmen. Aktuelle Methoden wie der mwerSegmenter richten maschinelle Ausgaben an Referenzen aus, indem sie die Wortfehlerrate minimieren, berücksichtigen jedoch keine semantischen Inhalte.

Diese Arbeit stellt zwei semantische Resegmentierungsansätze vor, die darauf ausgerichtet sind, diese Einschränkung zu beheben: einen, der Satz-Embeddings zur Ähnlichkeitsbewertung zwischen Segmenten verwendet, und einen anderen, der ein Sprachmodell für die direkte Resegmentierung einsetzt. Ziel ist es, die Segmentierungsqualität zu verbessern, um die Zuverlässigkeit automatischer Metriken und die Korrelation zwischen metrischen Bewertungen und menschlichen Urteilen zu erhöhen. Während die Sprachmodell-basierte Methode Potenzial zeigt, indem sie häufige Probleme von MWER vermeidet, ist sie inkonsistent und scheitert viele Übersetzungen korrekt zu segmentieren. Der Ansatz der semantischen Ähnlichkeit schneidet bei segmentbezogenen Korrelationen etwas schlechter ab als MWER, übertrifft jedoch oft MWER auf Dokumenten- und Systemeebene, insbesondere bei Metriken wie BERTScore. Darüber hinaus ermöglicht die semantische Segmentierung eine referenzfreie Resegmentierung und somit Segmentbewertungen ohne Referenz.

Insgesamt führen Verbesserungen gegenüber der MWER-Segmentierung zu moderaten Korrelationserhöhungen für beide Ansätze. Die maximale beobachtete Verbesserung beträgt etwa 0,05, wobei einzelne Ausreißer bei der COMET-Metrik höhere Werte erreichen.

Contents

List of Figures

List of Tables

1 Introduction

Speech translation systems convert spoken content from one language to another. To handle longer recordings, these systems typically segment the input into smaller units. In order to evaluate the quality of these systems, the output is compared to a reference created by human translators. However, these references are often segmented differently than the machine-generated translations.

To facilitate segment-by-segment evaluation, machine translation outputs are re-segmented to align with the reference. Current approaches, like mwerSegmenter, minimize the word error rate (WER) to align machine-generated segments with the reference.

The accuracy of automatic evaluation metrics is influenced by the quality of this re-segmentation. mwerSegmenter does not consider semantic content, and can make mistakes when the machine and reference translations are different. Such errors can weaken the correlation between automatic metrics and human judgments.

This thesis proposes two semantic re-segmentation methods, and evaluates their impact on the performance of automatic metrics, as well as segmentation quality in general. The goal is to produce a more accurate segmentation to improve the reliability of automatic metrics.

In cases where semantic segmentation performs better than MWER, the impact on correlation is moderate, with the maximum improvement being approximately 0.05. For bad translations and translations that include untranslated words, semantic segmentation can be more effective. It can also be used for reference-free re-segmentation, which could be useful in some cases.

1.1 Thesis Outline

This thesis is structured as follows: Following this introduction, Chapter 2 provides background information on the concepts and technologies used in this thesis. In Chapter 3 the re-segmentation approaches evaluated in this thesis are presented. Chapter 4 outlines the experimental setup, including information on the dataset, metrics, and evaluation methodology. The findings from the evaluation are analyzed in Chapter 5. Lastly, Chapter 6 concludes the thesis.

2 Background and Related Work

2.1 Tokenization and Embeddings

2.1.1 Tokenization

Tokens are the basic units of text used in natural language processing (NLP) [14]. Tokenization is the process of splitting text into these units. Tokens can be words, subwords, or characters, depending on the tokenization method used.

Word tokenization splits text into individual words. For example, the sentence "The fox jumps high" is tokenized into ["The", "fox", "jumps", "high"]. This method is simple but can struggle with compound words and contractions.

Subword tokenization breaks words into smaller units. This is useful for handling rare or unknown words. For example, "tokenization" might be tokenized into ["token", "ization"].

Each token is assigned a unique integer ID from a vocabulary. Unknown words are replaced with a special token.

2.1.2 Embeddings

Word embeddings are real-valued vector representations of words or tokens in a continuous vector space [9]. They are learned based on the context in which words appear. Words with similar meanings have vectors that are close in the vector space. This way embeddings capture semantic relationships between words and are used as input to neural networks in NLP tasks.

2.2 Transformers

The transformer is a neural network architecture designed to handle sequential data, such as text [29]. It utilizes a mechanism called self-attention, which enables it to capture long-range dependencies between tokens in a sequence [3].

The original Transformer architecture, depicted in Figure 2.1, consists of an encoder and a decoder [28, 4]. The encoder processes the input sequence. It consists of multiple identical layers where each layer contains a multi-head attention mechanism and a feedforward

neural network. The decoder generates the output sequence based on the information encoded by the encoder and the previous output. It also consists of multiple identical layers similar to the encoder, but with an additional multi-head attention mechanism that attends to the encoder's output. The other attention mechanism in the decoder is masked to prevent it from attending to future tokens. This masking, combined with offsetting the output embeddings by one position, ensures that the prediction for each new token only depends on previous tokens.

Because the transformer does not inherently consider word order, a positional encoding is added to the input embeddings to provide information about the position of each word in the sequence. These encodings are added to the input embeddings before feeding them into the self-attention layers.

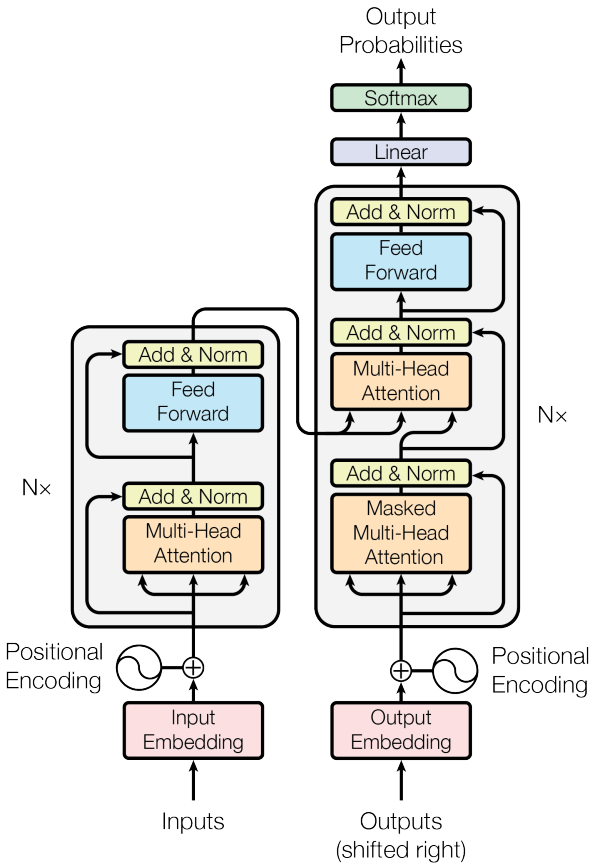


Figure 2.1: Original Transformer architecture [29]

The self-attention mechanism computes relationships between all tokens in a sequence. This provides direct connections between all tokens and enables the model to capture long-range dependencies. Each token is represented by a query Q, key K, and value vector V. The output is a weighted sum of the values, where the weights are determined by a softmax function applied on the similarity between the query and key vectors. This similarity is

computed as the dot product of the query and key vectors. Overall the attention scores are computed as follows:

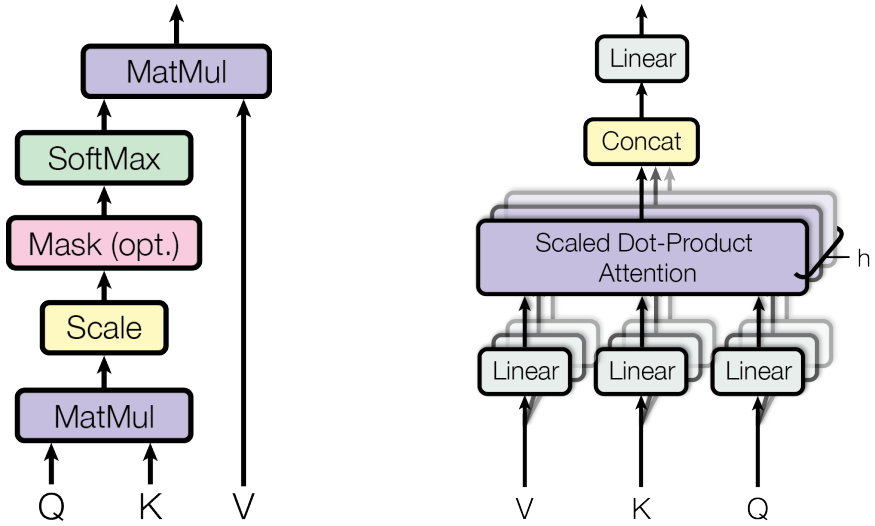
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V.$$

The scaling factor $\sqrt{d_k}$ ensures numerical stability when the dimensionality of the key vectors, d_k, is large. The whole process is illustrated in Figure 2.2a.

Multi-head attention extends self-attention by linearly projecting the queries, keys, and values h times with different, learned linear projections to d_k, d_k and d_v dimensions, respectively. Attention is computed in parallel for each subspace, and the outputs are concatenated:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O,$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$, and W^O is a learned output projection matrix. This process is illustrated in Figure 2.2b. Multi-head attention enables the model to simultaneously attend to information from different aspects of the input.



(a) Scaled Dot-Product Attention [29] **(b)** Multi-Head Attention [29]

Figure 2.2: Attention mechanisms of the Transformer

2.3 Large Language Models

Large language models (LLMs) are artificial neural networks trained on vast amounts of data to understand and generate human language. They are typically based on transformers to capture relationships between words in a context-aware manner.

2.3.1 Low-Rank Adaptation

Low-Rank Adaptation (LoRA) is a method for adapting pre-trained language models to new tasks or domains with limited data [8]. Instead of updating all model parameters, LoRA introduces task-specific low-rank matrices while keeping the original parameters frozen. This reduces the number of trainable parameters and computational requirements.

LoRA achieves this by decomposing weight updates into the product of two smaller matrices with low rank. These matrices have fewer parameters than the original weight matrices. This is particularly useful in cases where data is limited, as it reduces the risk of overfitting.

Another advantage of LoRA is its modularity. The low-rank matrices can be added or removed without altering the model's original structure. This makes it possible to adapt a single pre-trained model to multiple tasks by swapping in different adapters.

2.3.2 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a language model built on the transformer architecture [6]. It uses the encoder component of the original transformer architecture, as it is designed create deep representations of text rather than generate it.

Unlike models that process text in a single direction, BERT considers both preceding and following words in a sentence simultaneously, which allows it to capture richer contextual information. To achieve this, BERT uses a masked language model (MLM) training objective, where random tokens in the input are replaced with a mask token, and the model is trained to predict these masked tokens based on their context. Additionally, BERT is trained on a next sentence prediction (NSP) task, where it learns to predict if one sentence logically follows another. This aims to improve the model's understanding of inter-sentence relationships.

RoBERTa (Robustly optimized BERT approach) is a language model built on the foundation of BERT [12]. It retains the same architecture as BERT but changes the training methodology. While BERT uses static masking where mask positions are fixed during training, RoBERTa uses dynamic masking, where the mask positions are generated every time a sequence is fed to the model. The NSP task is also removed, as it did not have a positive impact on performance. RoBERTa is also trained with larger batch sizes and more data, which improves performance.

2.3.2.1 Sentence Embeddings

Sentence-BERT (SBERT) modifies a pre-trained BERT network with the goal of generating semantically meaningful sentence embeddings [23, 22]. It uses a siamese network architecture, where two identical BERT networks share weights and are fed two input sentences. The individual token embeddings for each sentence are passed through a mean pooling layer to generate a single sentence embedding. Then the model is trained to maximize the

similarity between embeddings of similar sentences and minimize the similarity between embeddings of dissimilar sentences. These sentence embeddings can be used for various tasks, such as semantic similarity and clustering.

This thesis uses both of the following multilingual sentence embedding models:

distiluse-base-multilingual-cased-v2[1], based on a distilled version of BERT [25]. This model includes an additional fully connected layer after the pooling layer to generate the final sentence embeddings.

jina-embeddings-v3[2], based on XLM-RoBERTa [27, 5]. This model includes a set of LoRA adapters for various tasks. For semantic similarity the "text-matching" LoRA is used.

2.3.3 Qwen2.5

Qwen2.5 is a series of generative large language models ranging from 0.5 to 72 billion parameters [17]. They use a transformer-based decoder-only architecture. Decoder-only models generate text sequentially, predicting the next token based on the preceding tokens.

This thesis uses the 1.5 and 7 billion parameter versions of the model. Both consist of 28 layers and have a maximum generated sequence length of 8k tokens. The 1.5 billion parameter version shares the weights of the input and output layers to reduce the number of parameters.

2.4 Translation Evaluation

2.4.1 Manual Evaluation

Manual evaluation of machine translation systems involves human annotators assessing the quality of translations. One common method is direct assessment (DA), where annotators rate translations on a continuous scale, based on predefined criteria [1, 7]. For source-based DA annotators are shown both the source text and the translated target text [26].

2.4.2 Automatic Evaluation

Automatic evaluation of machine translation systems assesses translation quality without the need for manual evaluation by human annotators. It uses algorithms and metrics to compare machine-generated translations against reference translations. The goal is to produce a quality score that closely correlates with human judgments of translation quality. The main benefit of automatic evaluation is much higher speed and scalability compared

[1] `https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2`
[2] `https://huggingface.co/jinaai/jina-embeddings-v3`

to manual evaluation. This allows researchers to quickly evaluate and compare different machine translation systems and configurations.

2.5 Evaluation Metrics

There are several types of evaluation metrics: Untrained metrics that compare a translation and reference directly, for example chrF uses character n-grams to calculate a score [16]. Metrics like BERTScore that use a pre-trained model to calculate embeddings and produce a score based on the similarity of those embeddings [30]. And trained metrics like COMET that learn to predict human judgments based on features from the machine translation [20].

2.5.1 Correlation with Human Judgments

The quality of an evaluation metric is often measured by its correlation with human judgments of translation quality. A high correlation indicates that the metric is effective at capturing the aspects of translation quality that are important to humans. Correlation can be calculated using various methods, such as the Pearson correlation coefficient and Spearman's rank correlation coefficient.

The Pearson correlation coefficient (ρ), measures the linear relationship between two sets of data X and Y. It quantifies how changes in one variable are associated with changes in the other. Values can range from -1 (perfect negative correlation) to 1 (perfect positive correlation). A value of 0 indicates no correlation. The formula for the Pearson correlation coefficient is:

$$\rho = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where:

- x_i and y_i are the data points of X and Y respectively,
- $\bar{x}$ and $\bar{y}$ are the means of X and Y,
- n is the number of data points.

Spearman's rank correlation coefficient r measures the monotonic relationship between two sets of data X and Y. It evaluates how well the relationship between two variables can be described using a monotonic function. It is defined as the Pearson correlation coefficient between the ranks of the data and can be calculated as

$$r = \rho[\mathrm{R}(X), \mathrm{R}(Y)]$$

where $\mathrm{R}(X)$ and $\mathrm{R}(Y)$ are the ranks of the data in X and Y respectively. While Pearson correlation only assesses linear relationships, Spearman's correlation can assesses monotonic relationships that are not necessarily linear. Like Pearson correlation, Spearman's correlation values range from -1 to 1.

2.5.2 chrF

The chrF metric evaluates the quality of machine translation output by comparing it to a reference translation using character level n-grams [16]. An n-gram is a contiguous sequence of n characters, not including spaces. For example, for $n = 2$, the word "machine" produces the bigrams "ma", "ac", "ch", "hi", "in", "ne". While it is possible to include word-level n-grams as well, this thesis only uses character-level n-grams.

To calculate a score, chrF uses $n = 6$ and $\beta = 2$ with the F-score formula:

$$\text{CHRF}_\beta = (1 + \beta^2) \cdot \frac{\text{CHRP} \cdot \text{CHRR}}{\beta^2 \cdot \text{CHRP} + \text{CHRR}}$$

Where:

- CHRP is the precision: the percentage of n-grams in the hypothesis that also appear in the reference.
- CHRR is the recall: the percentage of n-grams in the reference that also appear in the hypothesis.
- β adjusts the relative importance of recall and precision ($\beta = 1$ for equal weighting).

2.5.3 BERTScore

Instead of exact text matching, BERTScore is an evaluation metric that uses contextual embeddings to compute the similarity between a reference sentence and a candidate sentence [30]. Given a reference sentence $x = \langle x_1, \ldots, x_k \rangle$ and a candidate sentence $\hat{x} = \langle \hat{x}_1, \ldots, \hat{x}_l \rangle$, BERTScore creates contextual embeddings for each token using a pre-trained BERT model. It then computes the pairwise cosine similarity between the embeddings of each token in the candidate sentence and the reference sentence. Optionally, the result can be weighted with inverse document frequency scores, but this is not used in this thesis. Figure 2.3 illustrates the similarity computation.

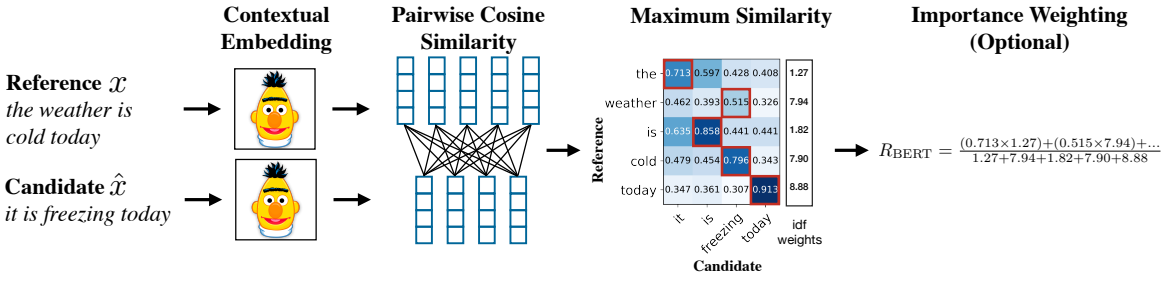


Figure 2.3: BERTScore calculation of the recall metric BERTR, greedy matching is highlighted in red [30].

To calculate the cosine similarity BERTScore uses pre-normalized tokens. Given a reference token x_i and a candidate token $\hat{x}_j$, the similarity is calculated as the inner product $x_i^T \hat{x}_j$.

Then each token in x is matched to a token in $\hat{x}$ to compute recall, and each token in $\hat{x}$ is matched to a token in x to compute precision. It uses greedy matching to maximize the similarity score, where each token is matched to the most similar token in the other sentence.

Similar to chrF, for a reference x and a candidate $\hat{x}$ the final score is calculated as an F-score using precision and recall, but with $\beta = 1$:

$$\text{BERTR} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \quad , \quad \text{BERTP} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \quad , \quad \text{BERTF} = 2 \cdot \frac{\text{BERTP} \cdot \text{BERTR}}{\text{BERTP} + \text{BERTR}} \quad .$$

2.5.4 COMET

COMET is a metric that directly predicts human quality judgments for translations using an estimator model [20, 18, 19]. Like BERTScore, it uses a pretrained model to calculate embeddings, in this case for the source, translation hypothesis, and reference. Unlike BERTScore, COMET does not use these embeddings to calculate an F-score. Instead, it uses a pooling layer to combine the embeddings into a single sentence embedding for each input. These embeddings are then concatenated with additional features and passed through a feed-forward regressor to predict the human quality judgment. This architecture is illustrated in Figure 2.4.

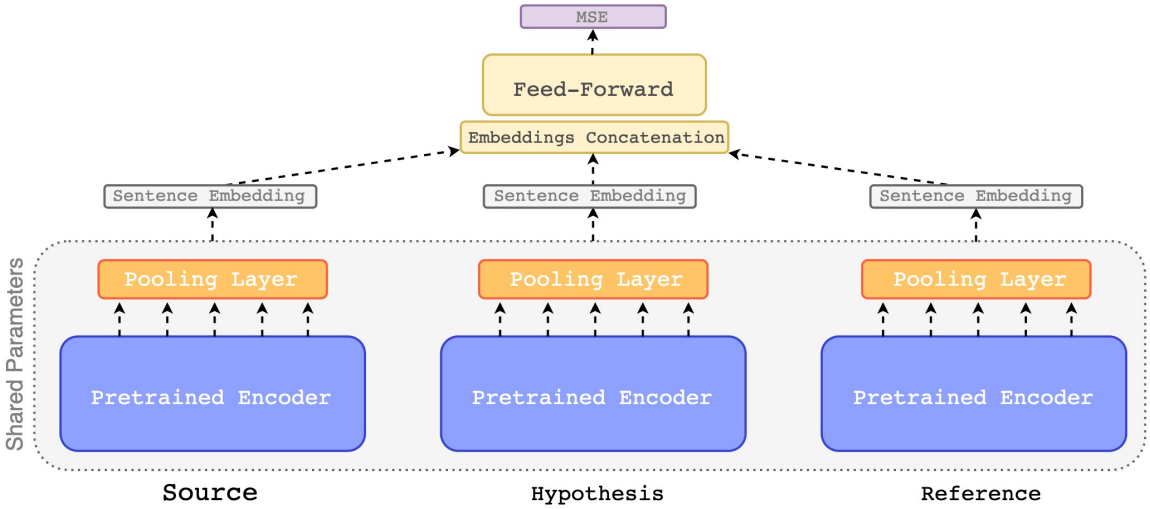


Figure 2.4: COMET estimator model architecture [20].

Given a d-dimensional sentence embedding for the source, the hypothesis, and the reference, it extracts the following combined features:

- Element-wise source product: $\boldsymbol{h} \odot \boldsymbol{s}$
- Element-wise reference product: $\boldsymbol{h} \odot \boldsymbol{r}$
- Absolute element-wise source difference: $|\boldsymbol{h} - \boldsymbol{s}|$
- Absolute element-wise reference difference: $|\boldsymbol{h} - \boldsymbol{r}|$

These combined features are then concatenated to the reference embedding $\boldsymbol{r}$ and hypothesis embedding $\boldsymbol{h}$ into a single vector $\boldsymbol{x} = [\boldsymbol{h}; \boldsymbol{r}; \boldsymbol{h} \odot \boldsymbol{s}; \boldsymbol{h} \odot \boldsymbol{r}; |\boldsymbol{h} - \boldsymbol{s}|; |\boldsymbol{h} - \boldsymbol{r}|]$ that serves as input to a feed-forward regressor. The features highlight the differences between embeddings in the semantic feature space. The source embedding $\boldsymbol{s}$ is not concatenated to the input vector, as the added value is negligible [20]. The model is then trained to minimize the mean squared error between the predicted scores and quality assessments like DA scores.

2.6 Resegmentation

Automatic translation systems typically produce translations in individual segments. In the case of text-to-text translation, these segments are often aligned with the sentence boundaries in the source text. However, content and number of these segments does not always align with the reference translations. Especially in the case of speech-to-text translation, sentence boundaries and word order can vary significantly. For example the speech recording could be segmented using a fixed-length window, while the reference translations are segmented based on the spoken content [2]. Additionally, the same spoken content can have multiple valid translations.

This makes the evaluation of individual segments difficult. To address this issue, the machine translation is re-segmented to align with the reference translations. In this process the complete translated text is split into segments that match their corresponding reference segments as closely as possible.

2.6.1 mwerSegmenter

MwerSegmenter is a tool for re-segmenting machine translation outputs [13]. It uses the Levenshtein edit distance to align words between the machine translation and reference translations [11, 15]. Instead of comparing sentence-by-sentence, it treats the entire machine translation output and reference translation as continuous streams of words.

It uses dynamic programming to find the optimal alignment path. For each word in the machine translation, it finds the best match in the reference translation through direct matches, substitutions, insertions, or deletions. This minimizes the total word-level error.

When aligning with multiple references, it selects the reference with the fewest errors for each segment. It normalizes reference lengths by padding shorter references with placeholders.

The segmenter identifies segment boundaries in the machine translation based on alignment results. It inserts boundaries where word alignments switch between segments in the reference so the re-segmented output mirrors the reference structure.

The final output is a re-segmented machine translation aligned with the reference sentence boundaries.

3 Approach

This chapter presents two different approaches for re-segmenting speech translation outputs. The first approach uses a semantic similarity model to align segments to a reference based on their content. The second approach uses an LLM to directly segment a translation output. Both approaches aim to re-segment mainly based on semantic content.

3.1 Semantic Similarity

The semantic similarity approach uses a pre-trained transformer model to compare the content of segments. It searches through possible segmentations of the machine translation output to find the best match to the reference segmentation. Each segmentation always has the same number of segments as the reference segmentation. To generate the different segmentations, the segment boundaries are initialized at the start of translation and recursively moved one word at a time.

The difference between a reference segment s_r and a potential segment of the machine translation s_t is calculated using the cosine similarity of their embeddings. The embeddings are generated by the transformer model for each token. Then the token embeddings are averaged to get the segment embedding. The cosine similarity can be calculated as the dot product of the segment embeddings divided by the product of their norms:

$$\text{cosine similarity} = S_C(s_r, s_t) = \frac{s_r \cdot s_t}{\|s_r\| \cdot \|s_t\|}$$

The best segmentation is found by maximizing the average similarity between each reference segment and the corresponding translation segment. Instead of averaging the token embeddings to create a segment embedding, it is also possible to maximize the BERTScore for each segment pair, which calculates the pairwise similarity of the token embeddings.

Searching through all segmentations for an entire translation is generally infeasible, as the number of possible segmentations grows exponentially with the number of segments. Instead, a lookahead window can be used to limit the number of segments considered at a time. Figure 3.1 illustrates this process with a window size of three segments.

The window moves over the reference segments one segment at a time. Only the start of the translation that matches the combined number of words of the reference segments within the window is considered. Once the optimal segmentation is found, the first segment is fixed. This segment is then removed from the translation, and the window advances by one

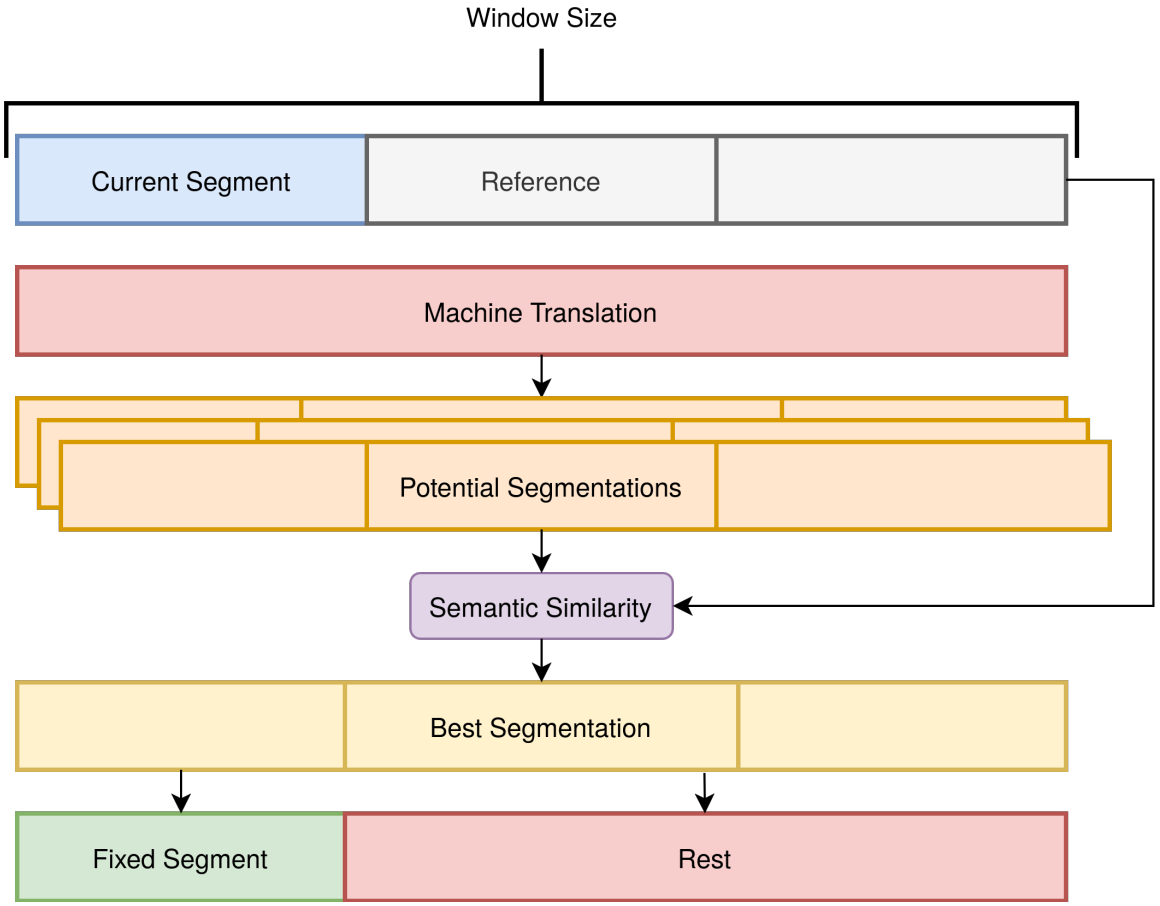


Figure 3.1: Semantic similarity approach

segment. This process continues iteratively, segmenting the translation one segment at a time. When the end of the window reaches the final segment, the remaining translation is segmented in the same step.

Another option to reduce the number of segmentations that need to be considered is to start with a larger step size. A larger step size means that the segment boundaries are moved by more than one word at a time. Then the best segmentation can be iteratively refined by decreasing the step size and only searching through similar segmentations. This approach is not guaranteed to find the optimal segmentation because the average similarity could have multiple local maxima. Moving a segment boundary by one word might decrease the similarity with the reference, but moving it further might increase it again. In practice however this only leads to a small difference in the final segmentation. For the data tested in this thesis, the difference between an initial step size of 8 and 1 is less than 1%.

It is also possible to use the segment boundaries from the previous iteration as a starting point for the next one. Because these segment boundaries should already be close to the optimal segmentation, they can be used to reduce the search space. Only the boundary for the last segment needs to be found from scratch.

3.2 Large Language Model

LLMs can also be used for segmenting translations. This approach leverages the ability of the model to understand and generate text.

The model is instructed to segment a machine translation to match the reference segmentation. It receives the machine translation as continuous text and the reference segments separated by newline characters. It then outputs a segmented machine translation separated by newline characters.

The produced segments are not guaranteed to match the original translation exactly. Occasionally, the model might change the spelling of a word or change the capitalization. To avoid these changes affecting the final segmentation, the model output is not used directly. Instead, the number of tokens in each segment is counted, and the original translation is segmented accordingly. This could lead to an incorrect segmentation if the model adds or removes words from the original translation. But it ensures that the content of the segments is the same as the original translation.

Similar to the semantic similarity method, the model only receives a limited number of segments at a time. The length of the machine translation the model receives is also matched to the total number of words in the reference segments. Limiting the segments keeps the number of segments consistent, regardless of the total number of reference segments. After fine-tuning on re-segmented data, this helps the model always return the correct number of segments. It also ensures that the resulting segmentation is within the maximum generated token limit of the model, and limits the amount of VRAM the model uses.

The fine-tuning data consists of multiple sets of segmented machine translations and reference translations. Each set has the same fixed number of segments. The model receives the concatenated machine translation and reference segments as input, and is trained to output the segmented version of the machine translation. The concatenated machine translation is trimmed or extended to match the total number of words in the reference segments. This matches the input the model will receive during inference.

4 Experiment Setup

No human-created reference re-segmentation is available for evaluating machine-translated texts. Instead, re-segmentation methods are assessed using automatic metrics, as well as qualitative analysis.

The re-segmentation methods evaluated include semantic sentence similarity based on the approach described in Chapter 3 and Minimum Word Error Rate (MWER) segmentation. For MWER the segmentation provided in the dataset is used. It was created using the mwerSegmenter tool, which minimizes word-level alignment errors between the translated output and the reference segmentation [13].

For similarity-based segmentation, the MWER segments are first joined to form a single text. This text is then re-segmented again to match the reference segmentation.

To calculate the similarity between segments, the Sentence Transformers library is used.[1] The specific model used for generating sentence embeddings is distiluse-base-multilingual-cased-v2, a multilingual transformer model which supports 50 languages.[2] Similarity between embeddings is calculated using cosine similarity.

4.1 Dataset

The dataset used for evaluation is the IWSLT 2023 dataset, which consists of multilingual translation tasks [26].[3] It includes both machine translations and reference translations from English to German (en-de), Japanese (en-ja), and Chinese (en-zh). Each translation is divided into segments that are manually annotated with DA scores. These scores serve as the ground truth for evaluating translation quality. To create these DA scores, annotators were instructed to disregard segmentation errors and focus solely on the quality of the translation.

The dataset consists of two domains: TED and ACL. This evaluation uses the ACL domain, which is based on presentations from ACL 2022. These presentations were delivered in English on technical topics by a variety of speakers [24]. There are five different presentations in total.

The ACL domain includes two evaluation tasks:

[1] `https://github.com/UKPLab/sentence-transformers`

[2] `https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2`

[3] `https://huggingface.co/datasets/IWSLT/da2023`

- **Offline**: Translating English speech into a single target language.
- **Multilingual**: Translating English speech into multiple target languages.

For both tasks, systems receive long-form audio as input. This audio must first be segmented before translation. After translation, the outputs must be re-segmented to align with the reference text for evaluation.

4.2 Metrics

Four different metrics are used to calculate the automatic segment scores: chrF, BERT, COMET and COMETKiwi. Each of these metrics calculates scores differently.

- **chrF**: Character-based n-gram F-score that measures the similarity between two segments [16].
- **BERTScore**: A pre-trained BERT model to calculate a similarity score by comparing contextual embeddings of tokens [30].
- **COMET**: A model directly trained to predict human scores for translations [20].
- **COMETKiwi**: A reference-free variant of COMET that uses the OpenKiwi quality estimation framework [21, 10].

4.3 Process

The evaluation process using automatic metrics consists of the following steps:

1. **Segmentation**: The machine translated output is re-segmented using MWER or semantic segmentation. For MWER segmentation, the segmentation provided in the dataset is used.
2. **Scoring**: Metric scores are calculated for each segment using different automatic metrics.
3. **Correlation**: The correlation between metric scores and human direct assessment (DA) scores is calculated using Pearson linear correlation (ρ) and Spearman rank correlation (r).
4. **Comparison**: The correlation values are compared to determine the effectiveness of each re-segmentation method.

The motivation behind this approach is that a better segmentation provides more accurate information for automatic metrics to calculate scores. This should result in scores that are closer to human DA scores, leading to higher correlation values. However, this is not always true because automatic metrics are not perfect.

4.3.1 Segmentation impact on correlation

The example in Table 4.1 shows that the best segmentation is not always the one with the highest correlation. Here the segmentation in the second column was specifically chosen to maximize the correlation of metric scores with human DA scores. In this case, with chrF as the metric, the best segmentation has a Pearson correlation of 0.23 while the optimized segmentation has 0.86. These values are calculated for the entire translated text, not just the segments shown in the table.

The reason for this discrepancy is that even with a perfect segmentation, the metric scores do not always align with human DA scores. For chrF this is partially because it is character-based and does not consider the semantics of the text. But even metrics like COMET frequently make mistakes. For example, in the entire IWSLT 2023 dataset, there are a total of 779 segments with a DA score of 0, but COMET does not score any segment as 0. The optimized segmentation changes these mistakes to still correlate with the DA scores. This is not useful for actually evaluating a translation, but it does show that a higher correlation does not always mean a better segmentation.

Best Segmentation	**Highest Correlation**	**Reference**
Was versteht ein Seh- und Sprachwandler, wenn er einen Highscore für dieses Bild und diesen Satz als übereinstimmend	Highscore für dieses Bild und diesen Satz als übereinstimmend und einen niedrigen für	Was hat ein Seh- und Sprach-Transformer verstanden, wenn er für dieses Bild und diesen Satz eine hohe Punktzahl zuweist?
und einen niedrigen für diesen einordnet?		Und hier eine niedrige Punktzahl?
Fokussieren sich Visions- und Sprachmodelle auf das Richtige,	diesen einordnet? Fokussieren sich Visions- und Sprachmodelle auf das Richtige,	Konzentrieren sich Seh- und Sprachmodelle auf die wesentlichen Punkte?

Table 4.1: Comparison of different segmentations.

Similarly, for every metric there are cases where a segmentation is subjectively better, but its correlation is slightly lower than that of the MWER segmentation. In such cases, the difference in correlation is generally less than 0.1, and on average the correlation is slightly higher for the better segmentation.

Also, the MWER segmentation, while it is not perfect, is consistently reasonable. This means a large difference in correlation in either direction indicates that the segmentation is incorrect. A very large positive jump is more likely to be due to a segmentation mistake that incidentally improves the correlation, like in the example above.

To verify that a worse segmentation decreases the correlation between metric scores and DA scores on average, noise is added to the MWER segmentation of en-de translations in the dataset. Each segment boundary is shifted in a random direction by up to 6 tokens, separated at white spaces.

Figure 4.1 shows the impact of noise on the Pearson correlation. The x-axis represents the number of tokens by which the segment boundaries are shifted.

The largest jump in correlation 0.11 and occurs with the comet metric. Here the correlation is 0.42 with no noise and 0.31 when the boundaries are shifted by 1 token.

All metrics show a similar trend, with the correlation decreasing as the noise increases. There is only one exception, where the correlation for BERTScore increases slightly going from 3 to 4 tokens of noise.

Overall this suggests that the quality of the segmentation has a significant and consistent impact on the correlation. While it is possible to construct a worse segmentation that increases the correlation, this is unlikely to happen by chance. Consistently improving on the MWER segmentation should therefore also improve the correlation on average.

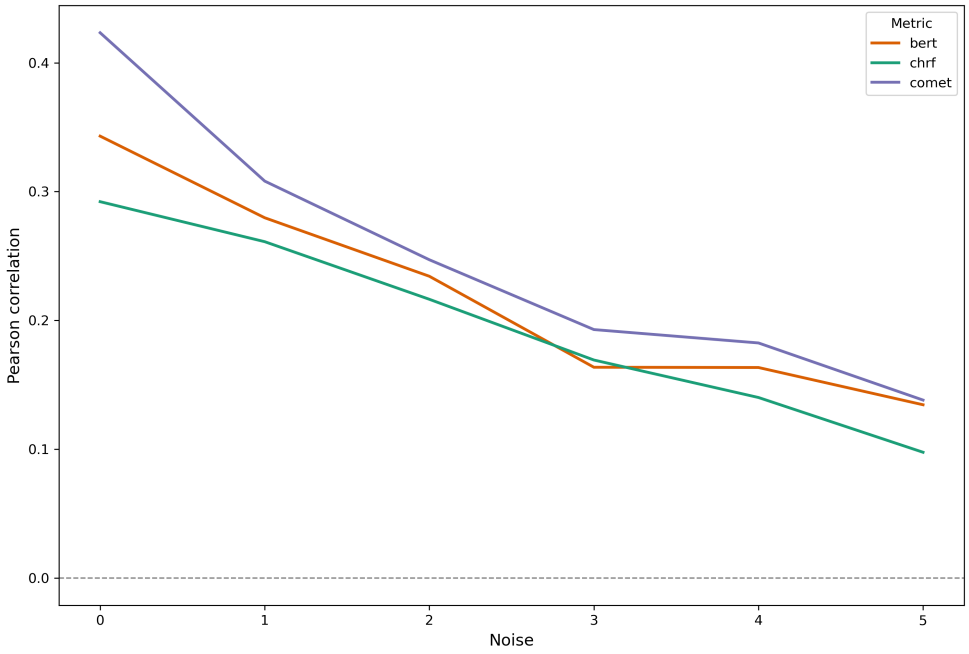


Figure 4.1: Impact of segmentation on pearson correlation

5 Results and Discussion

Table 5.1 presents the correlation between metric scores from different re-segmentation methods and DA scores across language pairs and tasks.

First the correlation of segment scores is calculated for each machine translation of a single ACL presentation individually. These values are then averaged within each category.

MWER segmentation generally achieves higher correlation values than similarity-based segmentation. For chrF and BERT, the correlation values are similar between the two methods, except for en-ja translations, where MWER performs better. The difference is most pronounced for the COMET metric, which is unexpected. Intuitively a semantic segmentation should be beneficial for a metric that is trained on human evaluations.

COMET is also more sensitive to segmentation changes, which can be seen in Figure 4.3.1. Because both chrF and BERTScore use an F-score to calculate the similarity between segments, the impact of small changes in segmentation is limited. COMET on the other hand uses a feed-forward neural network to predict human scores. This would explain why it has the largest difference in correlation. It is still strange that, compared to Figure 4.3.1 where the different metrics follow a similar trend, the correlation for chrF and BERTScore is almost unchanged for en-de translations. Especially BERTScore, which also evaluates segments based on semantic content, should be more similar to COMET. Maybe by segmenting exclusively based on semantic content, some information is lost that is important for the COMET metric.

				chrF		BERT		COMET	
Task	Lang.	Systems	Segmentation	ρ	r	ρ	r	ρ	r
Multi	en-de	3	Similarity	0.31	0.30	0.37	0.36	0.41	0.42
			MWER	0.31	0.30	0.37	0.35	0.45	0.44
Offline	en-de	7	Similarity	0.28	0.26	0.32	0.30	0.35	0.36
			MWER	0.28	0.26	0.33	0.32	0.41	0.41
Multi	en-ja	3	Similarity	0.29	0.29	0.38	0.37	0.40	0.40
			MWER	0.33	0.32	0.43	0.40	0.48	0.46
Multi	en-zh	3	Similarity	0.37	0.37	0.46	0.42	0.51	0.48
			MWER	0.39	0.37	0.48	0.44	0.55	0.52

Table 5.1: Correlation of individual segment metric scores and DA scores.

Table 5.2 shows the correlation between average metric scores and average human DA scores. Scores are averaged across all segments for each system in a category before calculating the correlation. This method aligns with the approach used in the IWSLT 2023 evaluation paper, and the MWER results are consistent with that paper [26].

Correlation values for average scores are higher than those for segment-by-segment results. This suggests that averaging metric scores improves accuracy compared to individual segment evaluations. However, the small number of systems per category limits the reliability of these correlation values.

When using average scores, similarity-based segmentation consistently shows higher correlation values than MWER segmentation. This contrasts with the segment-by-segment results, where MWER generally performed better. Even for the COMET metric, which showed the largest difference in favor of MWER at the segment level, similarity-based segmentation matches or outperforms MWER for average scores.

The cause of this discrepancy is unclear. It may be due to the small number of systems in each category, which can increase variability in the correlation results.

				chrF		BERT		COMET	
Task	Lang.	Systems	Segmentation	ρ	r	ρ	r	ρ	r
Multi	en-de	3	Similarity	0.99	1.0	0.99	1.0	1.0	1.0
			MWER	0.98	1.0	0.99	1.0	0.99	1.0
Offline	en-de	7	Similarity	0.95	0.75	0.98	0.75	0.99	0.96
			MWER	0.94	0.75	0.97	0.68	0.99	0.89
Multi	en-ja	3	Similarity	0.97	0.5	0.98	1.0	1.0	1.0
			MWER	0.97	0.5	0.98	1.0	1.0	1.0
Multi	en-zh	3	Similarity	1.0	1.0	1.0	1.0	0.99	1.0
			MWER	1.0	1.0	1.0	1.0	0.99	1.0

Table 5.2: Correlation of average metric scores with average DA scores per system.

In Table 5.3, the correlation between average metric scores and average DA scores is calculated for each individual presentation. Like in Table 5.2, the correlation values are very similar between the two segmentation methods, compared to the segment-by-segment correlations in Table 5.1. The largest difference in correlation is 0.03. However, the correlation values are more significant because there are more data points. For the multilingual task, instead of just three values when averaging scores per system, there are now 15 different values.

Compared to Table 5.2, semantic segmentation is not consistently better than MWER segmentation. For COMET it is only better for multilingual en-de translations, and for chrF it only slightly improves the Spearman correlation for en-ja translations. Otherwise, it is either slightly worse or the same as MWER segmentation for both metrics. The opposite

is true for BERTScore, where semantic segmentation performs better, except for en-zh translations.

				chrF		BERT		COMET	
Task	Lang.	Systems	Segmentation	ρ	r	ρ	r	ρ	r
Multi	en-de	3	Similarity	0.95	0.83	0.97	0.90	0.97	0.91
			MWER	0.96	0.85	0.96	0.87	0.96	0.90
Offline	en-de	7	Similarity	0.94	0.75	0.97	0.86	0.95	0.84
			MWER	0.94	0.77	0.96	0.83	0.96	0.86
Multi	en-ja	3	Similarity	0.73	0.64	0.87	0.82	0.96	0.89
			MWER	0.73	0.62	0.86	0.80	0.96	0.89
Multi	en-zh	3	Similarity	0.89	0.91	0.96	0.91	0.95	0.86
			MWER	0.90	0.92	0.97	0.94	0.96	0.87

Table 5.3: Correlation of average metric scores with average DA scores per individual presentation.

Figure 5.1 illustrates the difference in Pearson correlation between MWER segmentation and similarity-based segmentation for en-de translations. The difference is calculated as the Pearson correlation value for MWER segmentation minus the value for similarity-based segmentation. Each point represents the segment-level correlation for a single machine translation of a presentation. No two translations have the same average DA score. This means two points with the same average DA score are the same translation evaluated with a different metric.

The correlation values used are the same as those used in Table 5.1, with each value calculated for individual texts. For the COMET metric, MWER segmentation generally achieves higher correlation values than similarity-based segmentation. However, this difference is not consistent and shows greater variability across texts compared to chrF and BERT. This makes sense as COMET seems to be more sensitive to segmentation changes.

Similarity-based segmentation also performs worse for bad translations, although the sample size is small. Among the four translations with an average DA score below 0.6, MWER segmentation achieves higher correlation values across all metrics.

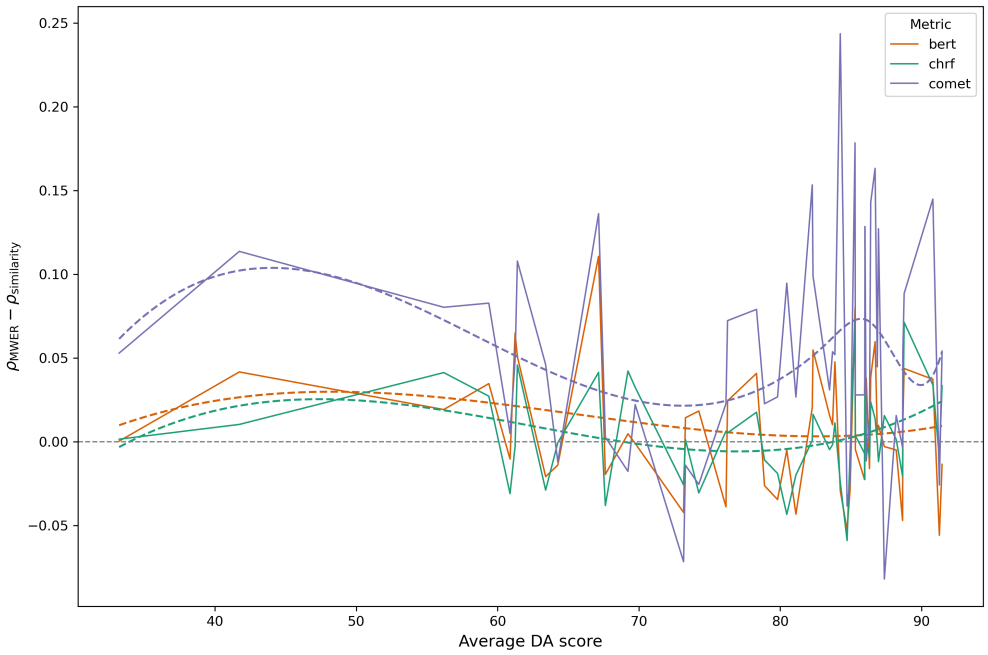


Figure 5.1: Difference in Pearson correlation between en-de translations segmented using MWER and sentence similarity. The dotted line is a smoothed spline over the corresponding points.

5.1 Reference-free re-segmentation

Some multilingual models create similar embeddings for the same words in different languages. This similarity can be used to re-segment a machine translated text without a reference, only a source transcription. The translation can then be evaluated using a reference-free metric like CometKiwi [21].

Table 5.4 shows the correlation between individual segment CometKiwi scores and DA scores for en-de translations. In this case the MWER segmentation is identical to the previous section, but the similarity-based segmentation only uses the source transcription.

While semantic segmentation performs worse than MWER segmentation, it is still comparable with a maximum difference of 0.07. This indicates that reference-free re-segmentation can be a viable approach for evaluating machine translations. It could be particularly useful when reference translations are unavailable or when assessing multilingual models that translate into many languages.

Task	Lang.	Systems	Segmentation	ρ	r
Multi	en-de	3	Similarity	0.44	0.44
			MWER	0.49	0.46
Offline	en-de	7	Similarity	0.38	0.39
			MWER	0.45	0.45

Table 5.4: Correlation of individual segment CometKiwi scores and DA scores.

5.2 Combining different models

It is possible to combine different models and techniques by averaging their scores for each segment during re-segmentation. In this section jina-embeddings-v3 are used to calculate sentence similarity scores [27].

Table 5.5 shows the correlation results. On its own, the model performs slightly worse than MWER segmentation. Combining it with BERTScore improves the correlation for all metrics, with COMET showing the largest improvement. Apart from using a different model to generate embeddings, BERTScore also calculates similarity differently. This could also be the reason for the improvement in correlation.

				chrF		BERT		COMET	
Task	Lang.	Systems	Segmentation	ρ	r	ρ	r	ρ	r
Multi	en-de	3	MWER	0.31	0.30	0.37	0.35	0.45	0.44
			Similarity	0.30	0.30	0.37	0.36	0.41	0.42
			+ BERTScore	0.30	0.30	0.37	0.36	0.43	0.44
			+ Source	0.31	0.30	0.37	0.36	0.45	0.45
			+ Punctuation	0.31	0.30	0.38	0.37	0.45	0.46
No Outlier: Multi	en-de	3	MWER	0.31	0.32	0.38	0.37	0.46	0.46
			Similarity	0.31	0.31	0.39	0.38	0.43	0.44
			+ BERTScore	0.31	0.32	0.38	0.38	0.43	0.44
			+ Source	0.31	0.32	0.38	0.38	0.44	0.45
			+ Punctuation	0.31	0.32	0.38	0.38	0.46	0.46

Table 5.5: Correlation of individual segment metric scores and DA scores.

Using the source transcription in addition to the reference translation and aligning segments to punctuation also improves the segmentation slightly. Segments are only aligned to punctuation if the resulting segmentation is within the top three best semantic segmentations for every segmentation step.

Overall this combination significantly improves the correlation for the COMET metric, which is now slightly higher than MWER segmentation. BERTScore and chrF on the other hand are almost unchanged.

However, a significant part of this improvement for COMET is due to the performance of semantic segmentation on a single translation of an ACL presentation. Figure 5.2 illustrates the difference in Pearson correlation between MWER segmentation and the combined model approach for all en-de translations. The biggest improvement is observed with moderately good translations. But except for one translation, the largest improvement in correlation compared to MWER is still only 0.05.

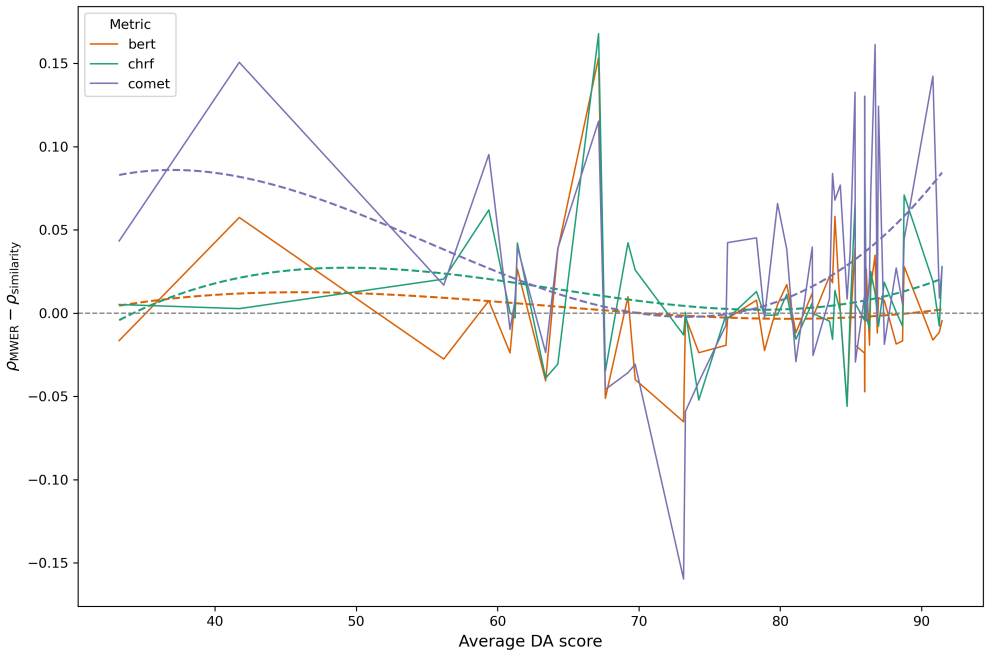


Figure 5.2: Difference in Pearson correlation between en-de translations segmented using MWER and the final ensemble from Table 5.5. The dotted line is a smoothed spline over the corresponding points.

The second part of Table 5.5 shows the results when removing this translation from the data. The initial correlation for both MWER and semantic segmentation is now slightly higher. Adding BERTScore to the semantic segmentation also does not improve the correlation anymore, and even slightly decreases the Pearson correlation for BERT. The source transcription does still improve the correlation for comet, but also not by much.

The removed translation contains multiple instances of untranslated words, as well as made-up or misspelled words that are phonetically similar to the original English words. This could explain why semantic segmentation performs better for this specific translation.

The embeddings for these words might be a lot more similar to the reference than the plain text. It also makes sense that referencing the source transcription would improve the segmentation for these types of errors. Why BERTScore also improves the segmentation in this case is less clear. It could be a difference in the model or the way the similarity is calculated.

Aligning segments to punctuation now causes the largest improvement in correlation. The final correlation scores without the outlier match those for MWER segmentation. This is still an improvement over the initial semantic segmentation, especially for the COMET metric.

5.3 Large Language Model

To test re-segmentation with a large language model, the 1.5B and 7B parameter versions of the Qwen2.5-7B[1] model are fine-tuned on TED data from the IWSLT 2023 dataset using low-rank adaptation [8]. This model was chosen because the larger 32B parameter version of the same model, which is too big to easily fine-tune, is already somewhat able to re-segment text without any fine-tuning. It is however not consistent and sometimes skips segments or just returns the reference segments. If the model is already inherently able to do re-segmentation, fine-tuning on a small amount of data could be enough to reinforce the task and structure of the data.

The model is fine-tuned on offline as well as simultaneous translation data from the TED portion of the IWSLT 2023 dataset. For the simultaneous translation task, translation models receive input that is already segmented according to the reference, so the resulting translations are also segmented correctly. For the offline task, the translations are re-segmented using mwerSegmenter.

The model quickly learns the correct structure of the input and output data. When it is trained exclusively on MWER segmented data, it also learns to include characteristic mistakes of the MWER segmentation, like shifting individual words to the wrong segment.

The 1.5B parameter version does learn to segment the translated text and the correct structure of the output, but it does not actually learn to associate the translation with the reference segments. Instead, it segments the translation blindly at points that seem reasonable. This works until the segmentation is slightly ambiguous, or the reference segments split a sentence in an unexpected way. If the model ever makes a mistake, the rest of the segmentation will go out of sync with the reference. For example if the model includes too much text in a segment, all following segments will be shifted forward by one. Even when later segments are clear or even identical to the reference, the model will not re-align them, because it does not actually associate the translation with the reference.

The 7B parameter version performs much better. For some translations, the model performs similarly to MWER segmentation, but it does not occasionally shift individual words to the

[1] `https://github.com/QwenLM/Qwen2.5`

wrong segment. This means in some cases the produced segmentation is actually perfect. But for bad translations, or bad sections in an otherwise good translation, the model has similar issues as the 1.5B parameter version. Although it is able to recover from a mistake, if later segments are clear, and will re-align the translation with the reference. When there are mistakes throughout the translation, the produced segmentation is completely wrong.

This is illustrated in Figure 5.3. The quality of segmentations produced by the 7B parameter model varies significantly between different translations. In most cases the segmentation is either very good or completely wrong.

For all instances where the correlation of COMET or BERT scores is within ~0.05 of the MWER segmentation, the segmentation produced by the model close to perfect. But in its current state the model is not useful in practice, because the output is very inconsistent. Usually re-segmentation is used to evaluate a translation when DA scores are not available. But without DA scores it not easy to determine if the segmentation is correct.

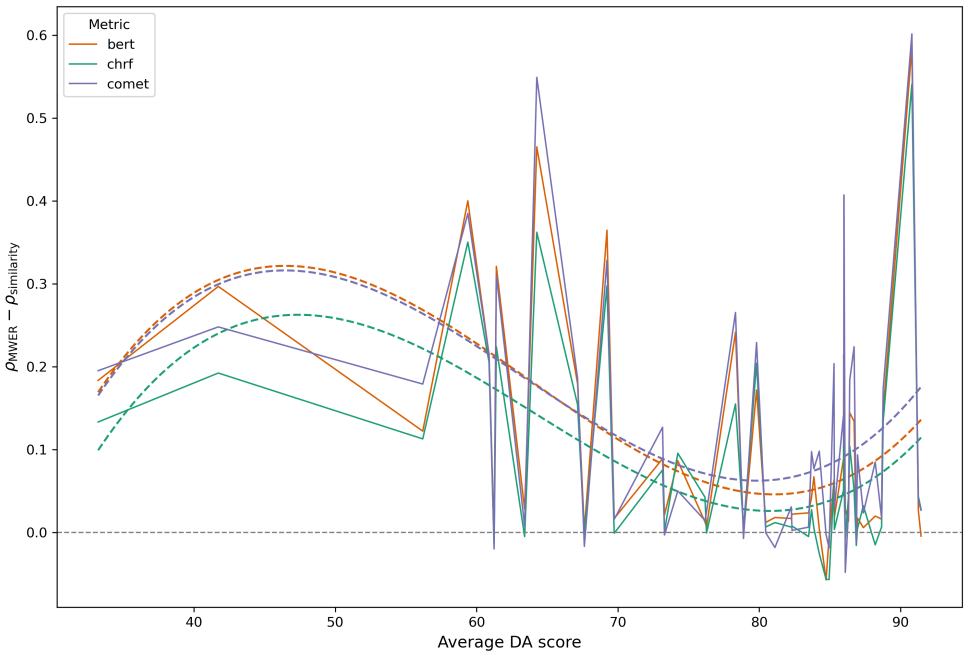


Figure 5.3: Difference in Pearson correlation between en-de translations segmented using MWER and a LLM. The dotted line is a smoothed spline over the corresponding points.

On average the LLM segmentation performs significantly worse than both MWER and semantic segmentation. Table 5.6 shows the segment-level correlation between metric scores and DA scores for the LLM segmentation. The correlation values are consistently lower than those for MWER and semantic segmentation across all metrics and tasks.

				chrF		BERT		COMET	
Task	Lang.	Systems	Segmentation	ρ	r	ρ	r	ρ	r
Multi	en-de	3	LLM	0.23	0.25	0.26	0.25	0.32	0.31
			Similarity	0.31	0.30	0.37	0.36	0.41	0.42
			MWER	0.31	0.30	0.37	0.35	0.45	0.44
Offline	en-de	7	LLM	0.21	0.21	0.23	0.24	0.30	0.32
			Similarity	0.28	0.26	0.32	0.30	0.35	0.36
			MWER	0.28	0.26	0.33	0.32	0.41	0.41

Table 5.6: Correlation of individual segment metric scores and DA scores.

This is likely due to the small amount of data used for fine-tuning, as well as the simplicity of the data. While there are multiple different translations, they are all based on the same source which contains only around 2000 segments.

The simultaneous translation data is also trivial to re-segment. Almost all segments are complete sentences starting with a capital letter and ending with punctuation. Adding noise to this data by changing or removing text in random segments also did not impact the results.

Using a larger version of the model could also improve the results. The 7B parameter version was not able to segment at all without any fine-tuning, but the 32B parameter version was in some cases. However, the 7B parameter version is already much slower than both MWER and similarity-based segmentation, and the 32B parameter version would be even slower. This might be acceptable for a single translation, but for a large dataset it could be impractical.

5.4 Qualitative

For bad translations, similarity-based segmentation can perform better than MWER segmentation. However, this does not actually lead to more accurate metric scores. In the example in Table 5.7, the human DA scores are 30 for the first segment and 0 for the second and third. For the second segment, COMET scores are 40 for MWER segmentation and 80 for similarity-based segmentation. So in this instance, a more accurate segmentation actually leads to a less accurate metric score.

It is also possible that the DA score for the second segment is incorrect. Because the MWER segmentation is incorrect for the second segment, it is possible that the part of the translation that actually corresponds to the reference was missed by the annotator.

MWER	Semantic	Reference
hohe Spezifität bedeutet, dass moder coo wetten leer sind nimmt in Fällen, in denen die res no annehmen leer sind hier waren das Ergebnis, da	hohe Spezifität bedeutet, dass moder coo wetten leer sind nimmt in Fällen, in denen die res no annehmen leer sind	Eine höhere Spezifität bedeutet, dass das Modell in Fällen, in denen die Versionshinweise leer sind, einen leeren Text korrekt ausgibt.
die Wüste enthält	hier waren das Ergebnis,	Hier sind die Resultate.
Madeses hat Barrieren etcetera wir auch berateda können Wüste, die ausschließt Angs	da die Wüste enthält Madeses hat Barrieren etcetera wir auch berateda können Wüste, die ausschließt	Da der Datensatz E-Mail-Adressen, Hash-Werte usw. enthält, haben wir auch den bereinigten Datensatz ausgewertet, der diese ausschließt.

Table 5.7: Comparison of MWER and semantic segmentation for a bad translation.

Similarity-based segmentation tends to produce less precise segment boundaries than MWER segmentation. This can be seen in the example in Table 5.8. Here individual words at segment boundaries are included in the wrong segment. While this does happen for MWER segmentation as well, it is less common.

For a good translation this can be improved by choosing a semantically close segmentation where segments end in punctuation. On the other hand if the translation is bad, it might not contain punctuation at all, and for more ambiguous segmentations this approach can make segmentation errors worse. Instead of shifting a single word to another segment, a whole sentence might be shifted.

MWER	Semantic	Reference
[...] in den Daten auszunutzen.	[...] in den Daten auszunutzen. Und	[...] in den Daten auszunutzen.
Und wir alle wissen, [...] und Abkürzungen nehmen.	wir alle wissen, [...] und Abkürzungen	Wir alle wissen, [...] und Abkürzungen nehmen.
Und wie gesagt, [...]	nehmen. Und wie gesagt, [...]	Wie wir schon erwähnten, [...]

Table 5.8: Comparison of MWER and semantic segmentation for a good translation.

For multilingual en-de data, which consists of 1247 segments, there are a total of 280 differences between MWER and semantic segmentation. Semantic segmentation in this

case uses jina embeddings and BERTScore with both the reference translation and source transcription to calculate similarity, as well as aligning segments to punctuation if the resulting segmentation is close to the best semantic segmentation. This is the same approach used in Section 5.2.

These differences are categorized as follows:

- Insignificant: 66 of the total differences are insignificant. These are cases where only a comma or a conjunction like "and" is included in a different segment.
- Minor: Cases where one or two words are included in a different segment, but they are not significant for the meaning of the segment.
- Moderate: Individual words that are important for the meaning of a segment are included in a different segment.
- Major: The content of a segment is completely different.

The distribution of these differences is shown in Figure 5.4. Most differences are minor, with semantic segmentation generally performing better. This is due to alignment to punctuation. Without this alignment there are ~50 additional differences and the MWER segmentation is better for the majority of minor differences.

For moderate and major differences, the performance is more balanced, though semantic segmentation is still slightly better. This is consistent with the results in Table 5.5 where this semantic approach has a slightly higher correlation with DA scores than MWER segmentation.

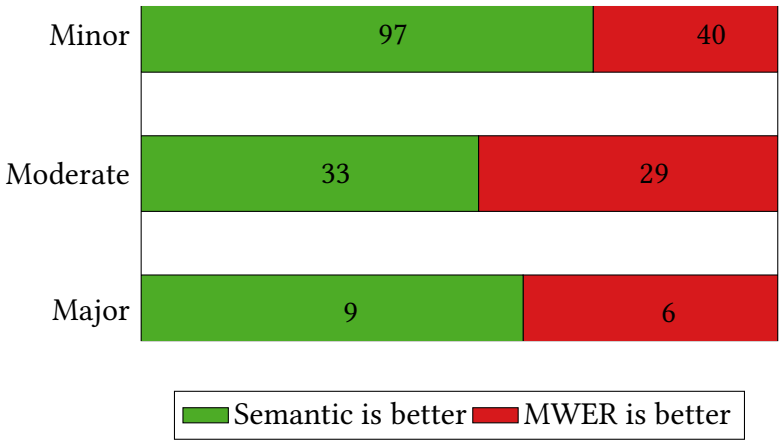


Figure 5.4: Differences between semantic and MWER segmentation for multilingual data.

6 Conclusion

This thesis presents two semantic re-segmentation approaches with the goal of improving the correlation between automatic metric scores and human DA scores for speech translations. The first approach uses sentence embeddings to calculate similarity between segments, while the second uses a LLM to re-segment the translations.

The LLM approach is not yet able to consistently produce accurate segmentations, but it shows potential for future work. In cases where the segmentation is good, it avoids the issues of MWER and similarity based segmentation, like shifting individual words to the wrong segment. In these cases this often results in perfect segmentations, which is rare for both MWER and semantic similarity, especially without optimizations like aligning segments to punctuation.

The semantic similarity approach performs worse than MWER segmentation on average for segment-level correlations. However, when averaging scores on a document or system level, semantic similarity frequently performs better, especially for the BERTScore metric. There are also cases where the system output includes untranslated words, which can be handled better by semantic segmentation.

There are also applications like reference-free re-segmentation, where semantic segmentation could be useful, since this is not possible with MWER segmentation. While the correlation is slightly lower than using MWER segmentation with a reference, it can still be a viable approach in some cases.

In general improving the segmentation over MWER seems to result in moderate improvements in correlation. For the translations evaluated in this thesis, the maximum observed improvement in correlation is 0.05, even in cases where the segmentation is close to perfect. There are outliers where the improvement is larger, but only for the COMET metric.

Finding an approach that consistently improves on MWER segmentation could still be valuable. Apart from the computational cost, using an LLM might be the best option, as it produces the most human-like segmentations. However, making the model more consistent likely requires a larger and more diverse dataset.

Another option could be to train a metric that accepts additional translation segments as context, while still only evaluating a single segment at a time. This would reduce the impact of segmentation errors. It could also address potential situations where a perfect segmentation is not possible due to differences in the sentence structure of the reference and machine translation.

Bibliography

[1] Farhad Akhbardeh et al. “Findings of the 2021 Conference on Machine Translation (WMT21)”. In: *Proceedings of the Sixth Conference on Machine Translation*. Ed. by Loic Barrault et al. Online: Association for Computational Linguistics, Nov. 2021, pp. 1–88. URL: https://aclanthology.org/2021.wmt-1.1/.

[2] Chantal Amrhein and Barry Haddow. “Don’t Discard Fixed-Window Audio Segmentation in Speech-to-Text Translation”. In: *Proceedings of the Seventh Conference on Machine Translation (WMT)*. Ed. by Philipp Koehn et al. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, Dec. 2022, pp. 203–219. URL: https://aclanthology.org/2022.wmt-1.13.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. Version Number: 7. 2014. DOI: 10.48550/ARXIV.1409.0473. URL: https://arxiv.org/abs/1409.0473 (visited on 01/04/2025).

[4] Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. Version Number: 3. 2014. DOI: 10.48550/ARXIV.1406.1078. URL: https://arxiv.org/abs/1406.1078 (visited on 01/04/2025).

[5] Alexis Conneau et al. *Unsupervised Cross-lingual Representation Learning at Scale*. Version Number: 2. 2019. DOI: 10.48550/ARXIV.1911.02116. URL: https://arxiv.org/abs/1911.02116 (visited on 01/08/2025).

[6] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Version Number: 2. 2018. DOI: 10.48550/ARXIV.1810.04805. URL: https://arxiv.org/abs/1810.04805 (visited on 01/01/2025).

[7] Yvette Graham et al. “Continuous Measurement Scales in Human Evaluation of Machine Translation”. In: *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. Ed. by Antonio Pareja-Lora, Maria Liakata, and Stefanie Dipper. Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 33–41. URL: https://aclanthology.org/W13-2305/.

[8] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. Oct. 16, 2021. DOI: 10.48550/arXiv.2106.09685. arXiv: 2106.09685[cs]. URL: http://arxiv.org/abs/2106.09685 (visited on 12/16/2024).

[9] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd. Aug. 20, 2024. URL: https://web.stanford.edu/~jurafsky/slp3/.

[10] Fábio Kepler et al. *OpenKiwi: An Open Source Framework for Quality Estimation*. Version Number: 2. 2019. DOI: 10.48550/ARXIV.1902.08646. URL: https://arxiv.org/abs/1902.08646 (visited on 01/09/2025).

[11] V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions and Reversals". In: *Soviet Physics Doklady* 10 (Feb. 1, 1966). ADS Bibcode: 1966SPhD...10..707L, p. 707. URL: https://ui.adsabs.harvard.edu/abs/1966SPhD...10..707L (visited on 01/01/2025).

[12] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. Version Number: 1. 2019. DOI: 10.48550/ARXIV.1907.11692. URL: https://arxiv.org/abs/1907.11692 (visited on 01/01/2025).

[13] Evgeny Matusov et al. "Evaluating Machine Translation Output with Automatic Sentence Segmentation". In: *Proceedings of the Second International Workshop on Spoken Language Translation*. Pittsburgh, Pennsylvania, USA, Oct. 25, 2005. URL: https://aclanthology.org/2005.iwslt-1.19.

[14] Sabrina J. Mielke et al. *Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP*. Version Number: 1. 2021. DOI: 10.48550/ARXIV.2112.10508. URL: https://arxiv.org/abs/2112.10508 (visited on 12/26/2024).

[15] Gonzalo Navarro. "A guided tour to approximate string matching". In: *ACM Computing Surveys* 33.1 (Mar. 2001), pp. 31–88. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/375360.375365. URL: https://dl.acm.org/doi/10.1145/375360.375365 (visited on 01/01/2025).

[16] Maja Popović. "chrF: character n-gram F-score for automatic MT evaluation". In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Proceedings of the Tenth Workshop on Statistical Machine Translation. Lisbon, Portugal: Association for Computational Linguistics, 2015, pp. 392–395. DOI: 10.18653/v1/W15-3049. URL: http://aclweb.org/anthology/W15-3049 (visited on 05/09/2024).

[17] Qwen et al. *Qwen2.5 Technical Report*. Version Number: 1. 2024. DOI: 10.48550/ARXIV.2412.15115. URL: https://arxiv.org/abs/2412.15115 (visited on 01/01/2025).

[18] Ricardo Rei et al. "Are References Really Needed? Unbabel-IST 2021 Submission for the Metrics Shared Task". In: *Proceedings of the Sixth Conference on Machine Translation*. Ed. by Loic Barrault et al. Online: Association for Computational Linguistics, Nov. 2021, pp. 1030–1040. URL: https://aclanthology.org/2021.wmt-1.111/.

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

[20] Ricardo Rei et al. *COMET: A Neural Framework for MT Evaluation*. Oct. 19, 2020. arXiv: 2009.09025[cs]. URL: http://arxiv.org/abs/2009.09025 (visited on 05/09/2024).

[21] Ricardo Rei et al. *CometKiwi: IST-Unbabel 2022 Submission for the Quality Estimation Shared Task*. Sept. 13, 2022. arXiv: 2209.06243[cs]. URL: http://arxiv.org/abs/2209.06243 (visited on 05/09/2024).

[22] Nils Reimers and Iryna Gurevych. *Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation*. Version Number: 2. 2020. DOI: 10.48550/ARXIV.2004.09813. URL: https://arxiv.org/abs/2004.09813 (visited on 01/01/2025).

[23] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. Version Number: 1. 2019. DOI: 10.48550/ARXIV.1908.10084. URL: https://arxiv.org/abs/1908.10084 (visited on 01/01/2025).

[24] Elizabeth Salesky et al. "Evaluating Multilingual Speech Translation under Realistic Conditions with Resegmentation and Terminology". In: *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*. Ed. by Elizabeth Salesky, Marcello Federico, and Marine Carpuat. Toronto, Canada (in-person and online): Association for Computational Linguistics, July 2023, pp. 62–78. DOI: 10.18653/v1/2023.iwslt-1.2. URL: https://aclanthology.org/2023.iwslt-1.2.

[25] Victor Sanh et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. Version Number: 4. 2019. DOI: 10.48550/ARXIV.1910.01108. URL: https://arxiv.org/abs/1910.01108 (visited on 01/08/2025).

[26] Matthias Sperber et al. *Evaluating the IWSLT2023 Speech Translation Tasks: Human Annotations, Automatic Metrics, and Segmentation*. June 6, 2024. arXiv: 2406.03881[cs]. URL: http://arxiv.org/abs/2406.03881 (visited on 11/15/2024).

[27] Saba Sturua et al. *jina-embeddings-v3: Multilingual Embeddings With Task LoRA*. Version Number: 3. 2024. DOI: 10.48550/ARXIV.2409.10173. URL: https://arxiv.org/abs/2409.10173 (visited on 01/01/2025).

[28] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. *Sequence to Sequence Learning with Neural Networks*. Version Number: 3. 2014. DOI: 10.48550/ARXIV.1409.3215. URL: https://arxiv.org/abs/1409.3215 (visited on 01/04/2025).

[29] Ashish Vaswani et al. *Attention Is All You Need*. Version Number: 7. 2017. DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762 (visited on 12/26/2024).

[30] Tianyi Zhang et al. *BERTScore: Evaluating Text Generation with BERT*. Feb. 24, 2020. arXiv: 1904.09675[cs]. URL: http://arxiv.org/abs/1904.09675 (visited on 11/16/2024).