English-Italian Machine Translation

Sona Mehdizade
Freie Universität Berlin
Berlin, Germany
sonam00@zedat.fu-berlin.de

ACM Reference Format:
Sona Mehdizade. . English-Italian Machine Translation. In *Proceedings of NLP course.* , 6 pages. https://doi.org/

Introduction

In this project, the Europarl parallel corpus, consisting of transcriptions from the proceedings of the European Parliament in English and Italian, is utilized. The aim of the given project is to develop a machine translation models to translate English text into Italian and vice versa. Various neural network-based modeling architectures are employed and their performance is evaluated.

Note: Opening the notebook with Jupyter on a local machine can cause validation errors. If this is the case, please access the Colab notebook with this link.

Task 1: EDA

For the initial exploratory analysis, since conducting EDA on the entire corpus was not feasible, 25% of the data was sampled to make it manageable with limited computational resources. Since the final data is extracted from this part, this subset is well-suited for understanding data which will be used for training of models.

As an initial data quality analysis, each instance was split into individual sentences to check sentence-level alignment between the English and Italian versions. In Figure 1,it is seen that 88.94% of instances had matching sentence counts, indicating strong alignment. However, some instances showed discrepancies due to differences in sentence structure or translation variations.

To maintain high-quality data, only instances with matching sentence counts were retained, ensuring one sentence per line. This step was crucial to reduce computational complexity and improve the model's learning efficiency by simplifying the training data and maintaning its consistency.

After processing, the final dataset contains 412,611 entries, each with one sentence pair per line.

1.1 Sentence Length Analysis

After defining the data structure and ensuring quality, the first analysis compared the average length of sentences in English and Italian using word count and character count metrics to understand their length differences.

According to Table 1, the analysis reveals that English sentences have slightly more words but fewer characters than Italian sentences. This suggests that English uses more, shorter words, while Italian uses longer words on average. These differences are likely due to linguistic characteristics: Italian often employs longer words and more elaborate grammatical structures, whereas English tends to use shorter words in greater quantity to express similar ideas.

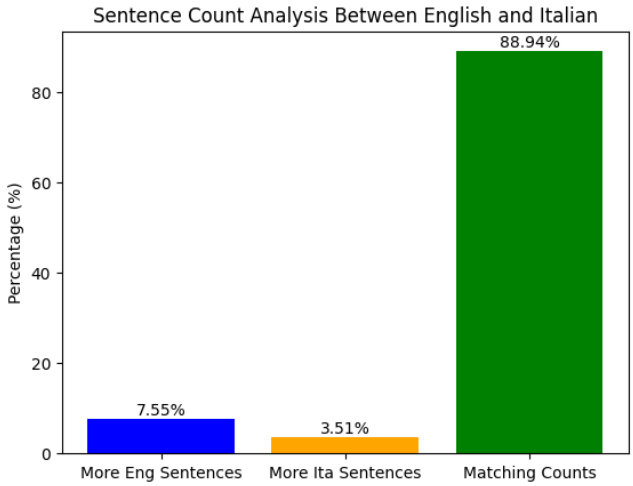

Figure 1: Sentence Count Analysis between English and Italian

Table 1: Average Sentence Length in English and Italian

Language	Mean Word Count	Mean Character Count
English	25.12	149.61
Italian	24.24	162.11

Next, word-level alignment between English and Italian sentences are evaluated. The analysis results in Figure 2 shows that English sentences are longer than Italian sentences in 52.98% of cases, while Italian sentences are longer in 35.30% of cases. In 11.72% of cases, the sentence lengths are equal. This discrepancy highlights the inherent structural differences between the two languages, with English generally having longer sentences overall.

1.2 Vocabulary Analysis

Continuing our analysis, the vocabulary richness of both languages is evaluated by counting the number of unique words in the English and Italian corpora.

According to Figure 3, the Italian corpus contains significantly more unique words (206,585 words) compared to the English corpus (138,114 words). This suggests that Italian has a richer vocabulary in the context of the Europarl proceedings. As a Romance language, Italian often has more word variations due to conjugations and

NLP course, July 2024, Berlin, Germany
. ACM ISBN
https://doi.org/

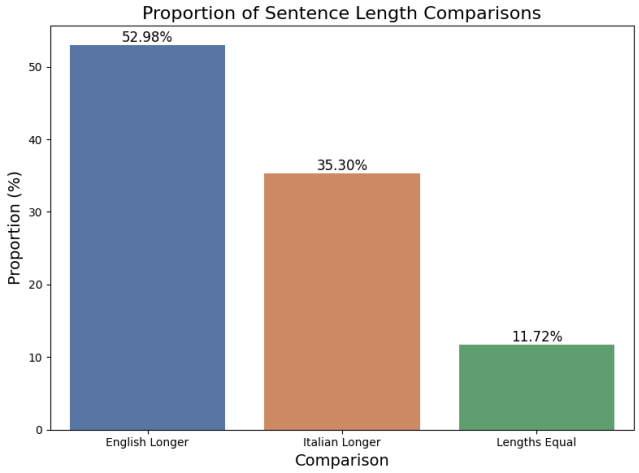

Figure 2: Word-level Sentence Length Comparison

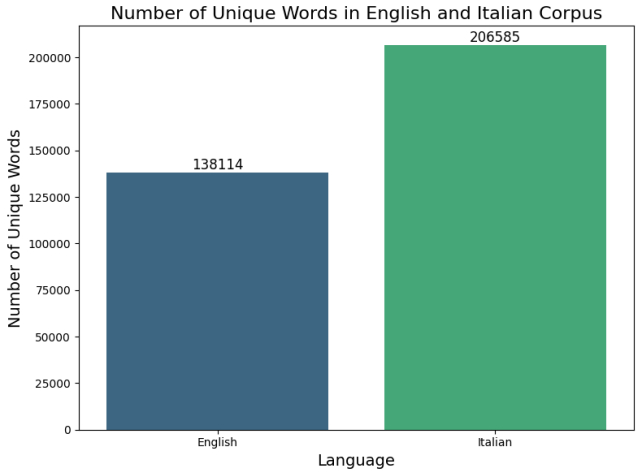

Figure 3: Number of Unique Words in English and Italian Corpus

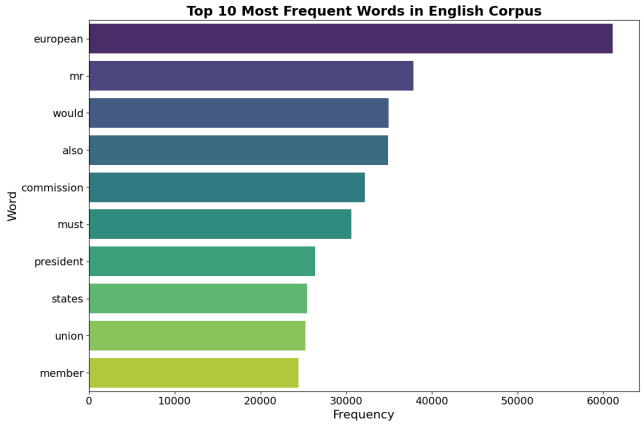

Figure 4: Top 10 Most Frequent Words in English

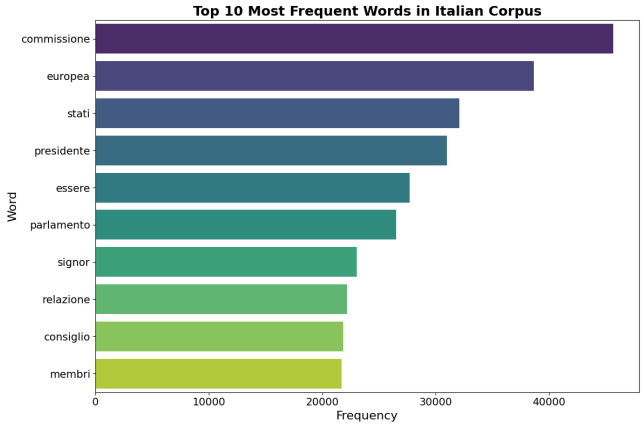

Figure 5: Top 10 Most Frequent Words in Italian

grammatical agreements, contributing to a higher count of unique words.

Next, the word frequency distributions in both languages were analyzed after removing stopwords and punctuation to identify the most common words.

The most frequent words in both languages, as shown in Figures 4 and 5, reflect key terms related to the European Parliament, such as "european", "commission", "president", and "member" in English, and "commissione", "europea", "presidente", and "membri" in Italian. This consistency indicates that the topics discussed in the Europarl proceedings focus on European governance and institutional terminology.

This analysis also gives us an alert about the presence of highly frequent words which may pose potential issues like model bias and overfitting, in which the model tend to predict frequent words more often, reducing output diversity by neglecting less frequent but important words.To address these issues, overfitting prevention techniques like regularization and dropout need to be implemented to ensure the model generalizes well.

Task 2: Preprocessing

For preprocessing, several general steps were applied to both languages:

(1) **Lowercasing**: Converted all text to lowercase to maintain uniformity and reduce vocabulary complexity.
(2) **Removing XML Tags and Empty Lines**: Removed lines starting with XML tags and any empty lines to clean the dataset.
(3) **Whitespace Normalization**: Removed extra spaces to ensure proper tokenization.
(4) **Punctuation Removal**: Removed all punctuation to simplify text data and reduce complexity.
(5) **Removing Duplicates**: Removed 11,874 duplicate instances to ensure data quality and consistency.

Special preprocessing steps for English included:

(1) **Removing Special Characters**: Removed all special characters except letters and numbers to focus on meaningful text content.
(2) **Decontracting**: Expanded contractions to their full forms to ensure uniformity in text representation. For example, "can't" was expanded to "cannot," and "won't" to "will not."

Special preprocessing steps for Italian included:

(1) **Specific Character Removal**: Removed specific special characters commonly found in Italian text (e.g., `[$);'%:,.`) while preserving essential diacritical marks specific for Italian language (e.g., à, è, ì) to maintain correct spelling and meaning.
(2) **Dash Replacement**: In Italian, dashes are occasionally used to connect words, such as in "pittore-scultore". Replacing dashes with spaces ensures that each word is handled separately, improving tokenization and the model's ability to process each component of compound terms accurately.

Certain techniques were not applied for the following reasons:

(1) **Removing Numbers**: Numbers were retained as they carry important information for translation (e.g., dates, quantities).
(2) **Stemming and Lemmatization**: These techniques were not applied as they reduce words to their base or root forms, potentially losing nuanced meanings. Additionally, stemming and lemmatization tools are typically language-specific, and ensuring accurate application for both languages could introduce errors.
(3) **Stopwords Removal**: Stopwords were not removed to maintain the coherence and contextual integrity of translated sentences.

To determine the optimal maximum length for padding, sentence lengths were analyzed. Based on the distribution provided in Table 2, a max length of 70 words covers the 99th percentile for both languages.

Table 2: Sentence Length Percentiles

Percentile	Italian	English
0th	1	1
10th	9	10
20th	13	13
30th	16	17
40th	19	20
50th	22	23
60th	26	26
70th	30	30
80th	35	35
90th	43	44
95th	51	52
99th	69	70
100th	416	435

Then, according to Table 2, it was decided that truncating sentences longer than 70 words could lead to a loss of context and important information, as word order and content might differ between languages. Long sentences often contain multiple subsentences, which might be reordered differently in the target language. This can cause misalignment and loss of meaning, as the remaining part of the sentence may not correspond correctly between languages. To maintain the integrity of translation pairs and ensure accurate learning, sentences longer than 70 words were removed instead of truncated.

After completing all preprocessing steps, the dataframe contained 412,478 instances. Due to computational limitations, only the first 200,000 samples were kept. After removing outliers based on word count, 197,678 sentence pairs remained in the final version.

Task 3: Neural Machine Translation

In this task, the following steps were performed to prepare the data for training a Neural Machine Translation model:

(1) **Adding Start and End Tokens**: `<start>` and `<end>` tokens were added to the English and Italian sentences to clearly mark the beginning and end of each sentence.
(2) **Loading Pre-trained Embeddings**: Various pre-trained embeddings were loaded to compare their impact on model performance. These included Word2Vec and FastText for both English and Italian, and GloVe for English only (GloVe embeddings for Italian were not available). Embedding matrices were created from the pre-trained embeddings for use in the model.
(3) **Splitting Data**: The dataset was split into training, validation, and test sets. 20% of the data was used as the test set, with the remaining data split further into training and validation sets as presented in Table 3. For evaluation, due to limited computational power, only a randomly sampled subset of 1500 instances from the test set is utilized. All the models are evaluated on this same subset of the test data.

Table 3: Dataset Split

Dataset	Sample Count
Training	118,606
Validation	39,536
Test	39,536

(4) **Tokenization**: Tokenizers were created for both English and Italian text, converting sentences into sequences of numbers. An out-of-vocabulary token ('UNK') was defined to handle words not present in the training vocabulary.
(5) **Preparing Encoder and Decoder Inputs/Outputs**: Sequences were padded with the max length found in the previous task based on percentiles to ensure consistent lengths, and input and output sequences were prepared for both the encoder and decoder.

The encoder-decoder architecture designed for this task includes specific layers and components for effective translation models. The architecture is depicted in Figure 6 below:

Encoder:

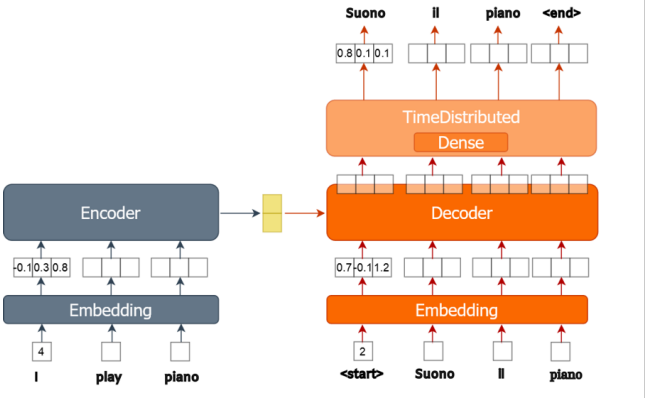

Figure 6: The utilized Encoder-Decoder Architecture
Diagram modified from original source of Datacamp

- **Embedding Layer:** Converts input tokens into dense vectors of fixed size, capturing semantic meaning and reducing dimensionality.
- **Bidirectional LSTM:**Captures information from past and future contexts, enhancing the model's ability to understand the full context, crucial for translation tasks where the meaning of a word depends on surrounding words.
- **Concatenation of States:** The forward and backward states of the Bidirectional LSTM are concatenated to form the final states.

Decoder:

- **Embedding Layer:** Similar to the encoder, this layer converts input tokens into dense vectors, maintaining consistency in how word representations are handled.
- **LSTM Layer:** Generates the output sequence one token at a time, maintaining long-term dependencies for coherent and contextually relevant translations.
- **Dense Layer with Softmax Activation:** Outputs a probability distribution over the vocabulary for each time step, predicting the next word in the sequence.
- **TimeDistributed Layer:** Ensures that the dense layer processes each time step separately, maintaining the sequential nature of the output.

Moreover, dropout layers are added to both the encoder and decoder to prevent overfitting, making the model less sensitive to specific weights.

Considering hyperparameters, the models used an embedding size of 300 for the embedding vectors and an LSTM size of 256 for each LSTM layer. The batch size is set to 64 (for attention-based, 256), and the learning rate for the Adam optimizer is 0.002. The model is trained for up to 10 epochs, with early stopping based on validation loss to prevent overfitting.

Overall, this architecture follows well-established practices in neural machine translation, using components proven effective in similar tasks.

To evaluate the performance of the models, two metrics were used: BLEU and METEOR. BLEU measures the precision of n-grams in the generated translation compared to reference translations, while METEOR considers both precision and recall, including synonym and stemming matches, providing a more nuanced evaluation of translation quality.

English to Italian translation model

The results of the English to Italian model with different embeddings are as follows:

Table 4: Results of the English to Italian Model with Different Embeddings

Embedding	Avg BLEU	Avg METEOR
Word2Vec	0.0111	0.1714
GloVe	0.0115	0.1786
FastText	0.0136	0.1876

As GloVe embeddings for Italian were not available, Word2Vec embeddings were used for the decoder part of the encoder-decoder model in GloVe evaluations.

The comparison shows that FastText embeddings, developed by Facebook's AI Research (FAIR) lab, resulted in superior performance. This is likely due to the advanced resources and computational power available at Facebook, which resulted in more comprehensive and higher-quality word embeddings. In contrast, the Word2Vec embeddings for Italian were developed by the Machine Learning Laboratory (MLSPTlab) at Università della Campania in Italy, which may have had more limited computational resources.

Overall, the results are generally poor, likely due to the insufficient amount of data and the need to keep the model less complex because of computational constraints. With more data and a more advanced model, combined with sufficient training time and computational power, performance could improve significantly. Pre-trained transformer-based models, such as BERT or GPT, could also be very useful for this task.

Moreover, the evaluation of instances lengths relation with their corresponding BLEU scores revealed that longer sentences typically have lower scores. The simple encoder-decoder model struggles with long sequences, often repeating the same 2-4 words towards the end of the translation. This highlights a significant limitation of the basic encoder-decoder architecture, as it fails to maintain context over longer sentences, leading to inaccurate translations.

It is generally observed that sentences that are shorter and simpler tend to result in better translations by the model. This is because the model can maintain context and coherence more effectively in shorter sequences. Additionally, sentences with common vocabulary and straightforward grammatical structures are translated more accurately, as the model is better able to handle familiar patterns and words.

Italian to English translation model

By changing the input and target languages, models with different embeddings were trained similarly. The results are presented in Table 5 below:

Table 5: Results of the Italian to English Model with Different Embeddings

Embedding	Avg BLEU	Avg METEOR
Word2Vec	0.0107	0.1944
GloVe	0.0199	0.2236
FastText	0.0204	0.2223

As observed in the English to Italian model, FastText embeddings demonstrated the best performance for the Italian to English model as well. However, the results remain generally low.

Similar to the English to Italian model, the Italian to English model struggles with long sequence predictions. However,compared to the English to Italian models, the results for the Italian to English translation models are slightly better. One probable reason for this improvement is the difference in vocabulary size between the two languages. As discussed in the EDA part, the Italian corpus contains significantly more unique words (206,585) compared to the English corpus (138,114). Therefore, when predicting the next word for the case where English is the target language, there are fewer choices, making the task somewhat easier for the model due to reduced vocabulary complexity. Additionally, English has a more straightforward sentence structure, which might be easier for the model to learn and predict.

Character-based Models

Character-based models were trained, offering reduced computational complexity due to the vocabulary size being limited to the number of letters in the languages. Additionally, character-based models can handle morphological variations and predict words that did not exist in the training data.

For the character-based models, several adjustments were made compared to the word-based models:

(1) **Special Tokens**: Unique characters $\in$ and $\notin$ were used to represent the start and end of sequences.
(2) **Tokenization**: Performed at the character level, converting each character into tokens.
(3) **Padding**: Sequences were padded to the 95th percentile length (max_len=330) to ensure consistent input sizes.

The embedding dimension for character-based models was set to 64 to balance complexity and efficiency. Apart from the change in embedding dimension, the character-based models used the same architecture and hyperparameters as the word-based models.

Table 6: Results of Character-based Models

Model	Mode	Avg BLEU	Avg METEOR
Character-based	Eng to Ita	0.0114	0.1554
Character-based	Ita to Eng	0.0054	0.1479

Comparing the results in Table 6 to word-based models' results with fasttext embedding, it is observed that for both translation tasks, the character-based model performed worse than the word-based model, likely due to the lack of semantic content in character-level models. Word-based models performed better, possibly due to benefiting from pre-trained word vectors that capture semantic relationships and contextual information.

Task 4: Neural Machine Translation with Attention layer

To improve the models, the Bahdanau attention mechanism was introduced. This mechanism helps the model focus on different parts of the input sequence when generating each word in the output sequence, simplifying the handling of long sequences.

Since FastText embeddings showed superior performance in previous models, only FastText embeddings were used for the attention-based models.

The results for both modes are demonstrated in 7 and 8 below, including the previous best models' (with FastText embeddings) results without Attention for clear comparison.

Table 7: Results of the English to Italian Model with and without Attention

Model	Avg BLEU	Avg METEOR
Word-based without attention	0.0136	0.1876
Word-based with attention	0.0868	0.3290
Character-based without attention	0.0114	0.1554
Character-based with attention	0.0226	0.1885

Table 8: Results of the Italian to English Model with and without Attention

Model	Avg BLEU	Avg METEOR
Word-based without attention	0.0204	0.2223
Word-based with attention	0.1195	0.4178
Character-based without attention	0.0054	0.1479
Character-based with attention	0.0198	0.2001

From the results above, it is noticed that the attention mechanism significantly improved the performance of both word-based and character-based models. Word-based models with attention showed significant improvements, benefiting from the ability to capture long-term dependencies and semantic relationships. Character-based models also improved with attention, though the gains were more modest.

The attention plot visualizes how the model focuses on different parts of the input sequence when generating each word in the output sequence. The example shows the translation of "E quello che stiamo facendo oggi." to "what is what we are doing today".

For example, considering a few from the plot, the model initially focuses on "E" and "quello" when predicting "what," showing an attempt to understand the context. For predicting "we," the focus moves to "stiamo," correctly identifying the pronoun related to the action. For "are," the attention overlaps with "stiamo," reinforcing

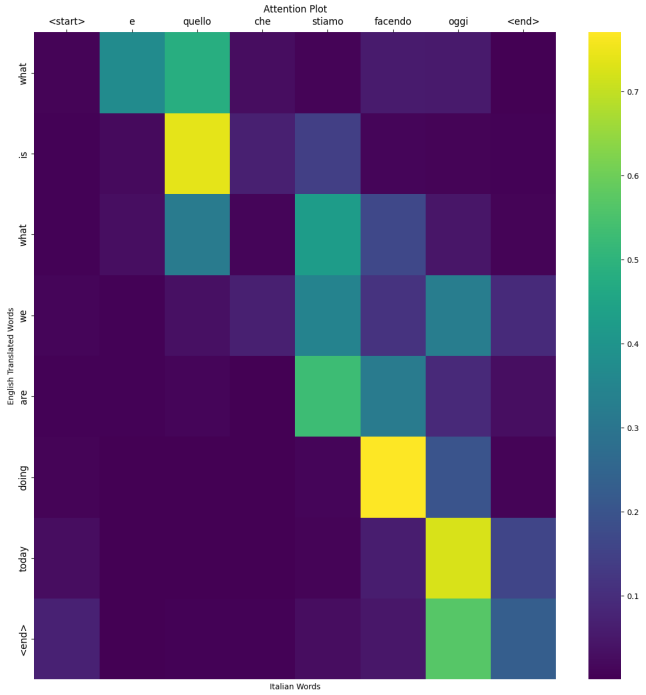

Figure 7: Attention weights of Italian to English model on sample sentence

the auxiliary verb. Finally, for "today," the focus is correctly placed on "oggi," indicating the model's understanding of the temporal aspect.

Bonus Task: Pivot Translation

Pivot translation is a technique used in machine translation where a pivot language, is used to translate between two languages that do not have sufficient direct parallel data. In this task, English is used as the pivot language to translate from German to Italian, leveraging the availability of more substantial German-English and English-Italian corpora.

The pivot translation process is clearly illustrated in Figure 8 below. Since we already have an English to Italian translation model, we need only to train a German to English model.

A German to English translation model was developed using the Bahdanau attention mechanism and FastText embeddings. The same preprocessing steps and model architecture used for English to Italian model were applied. The evaluation results of the model are given in Table 9 below:

Table 9: German to English Model Evaluation Results

Metric	Score
Avg BLEU	0.0637
Avg METEOR	0.3289

Moreover, the attention weights of German to English model on sample sentence can be observed in Figure 9.

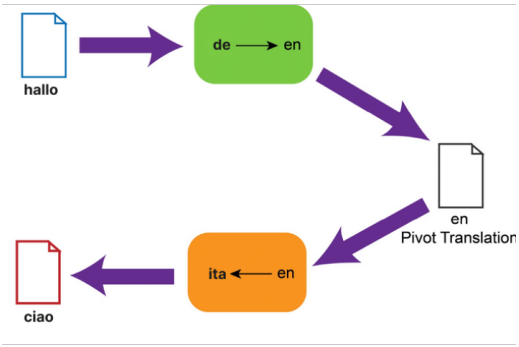

Figure 8: Pivot Translation procedure
Diagram modified from original source of Low-resource NMT

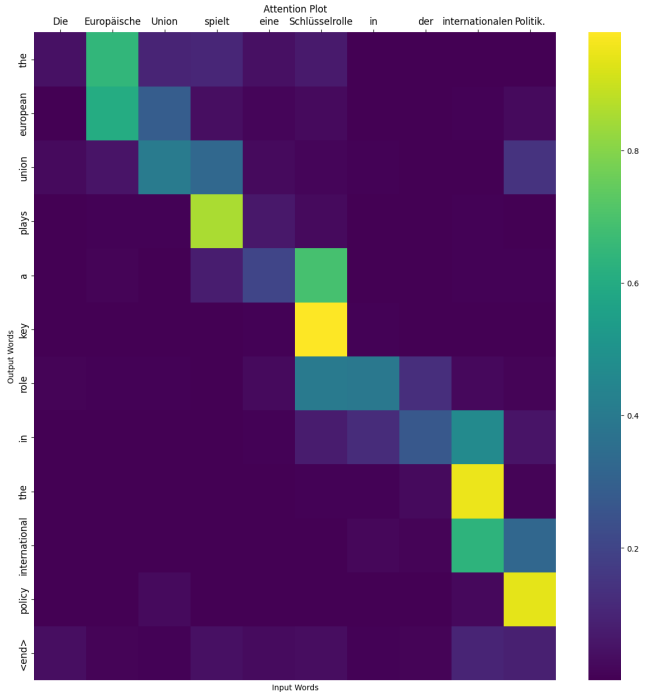

Figure 9: Attention weights of German to English model on sample sentence

So, after having two models, we apply pivot translation on one sample:

- **German sentence:** Die Europäische Union spielt eine Schlüsselrolle in der internationalen Politik.
- **Intermediate English translation:** The European Union plays a key role in international politics.
- **Final Italian translation:** L'Unione Europea svolge un ruolo chiave nella politica internazionale.

Overall, it is observed that the pivot translation approach demonstrated its effectiveness in leveraging English as an intermediary to translate German text into Italian, with attention mechanisms significantly enhancing translation quality.