Shahid Beheshti University
Data Science (2023-2024)

Recognizing Textual Entailment in SNLI dataset

Hossein Yahyaei
Department of Computer Science
Shahid Beheshti University
hossein.yb1210@gmail.com

Sarah Hosseini
Department of Computer Science
Shahid Beheshti University
husaynisara@gmail.com

Abstract

Textual entailment, the process of determining logical relationships between two sentences where one can be inferred from the other, holds significant importance in various natural language processing (NLP) applications. Recognizing and understanding entailment relationships is crucial for tasks such as question answering, information retrieval, and automated reasoning. In this study, we present a novel approach to textual entailment using two transformer-based models that are taught using the SNLI dataset.

The first model employs the premise-to-hypothesis (P->H) logic to assess the validity of the hypothesis (H) with respect to the premise (P). If H is deemed valid, it indicates an entailment relationship between P and H. The second model extends this framework by additionally considering the presence of information from P in H and vice versa as an indicator of a stronger entailment relationship. This second mechanism is more frequently used in recent works. However, entailment relationships are typically asymmetric, which means that P->H doesn't necessarily mean that H also entails P. Consequently, while the second model was initially expected to underperform compared to the first model, the experimental results confirmed this hypothesis.

Both models aim to address the textual entailment task, but they differ in their architectural designs. Our experimental results and performance evaluation shed light on the effectiveness and comparative performance of these models in the textual entailment domain.

1 Introduction

The textual entailment task, also known as natural language inference, is a fundamental problem in natural language processing (NLP) that aims to determine the logical relationship between a pair of text fragments. Given a premise (P) and a hypothesis (H), the task involves determining whether the hypothesis can be inferred from the premise, or if it contradicts or is unrelated to the information presented in the premise.

Solving the textual entailment task is crucial for various NLP applications, such as question answering, information retrieval, sentiment analysis, and machine translation. It plays a significant role in enabling machines to understand and reason about human language, advancing the development of intelligent systems that can comprehend and generate text.

However, the textual entailment task poses several challenges that must be addressed for accurate and robust performance. These challenges include:

(LateX template borrowed from NeurIPS 2019)

1. Ambiguity and Variability: Natural language is inherently ambiguous and variable, making it challenging to capture the precise meaning and the relationship between text fragments. Different wordings, lexical choices, and sentence structures can result in different interpretations, requiring models to handle this variability effectively.

2. Semantic Understanding: Understanding the meaning and semantic relationship between text fragments is essential for accurate textual entailment. Models must capture the nuances and context-specific information embedded within the text to determine if the hypothesis is entailed by or contradicts the premise.

3. Syntactic and Structural Dependencies: Textual entailment often requires capturing syntactic and structural dependencies between words and phrases. Models need to understand how the grammatical structure and word order influence the logical relationship between the premise and hypothesis.

4. World Knowledge and Reasoning: Effective textual entailment models should be capable of leveraging world knowledge and reasoning abilities to make accurate inferences. Reasoning over common-sense knowledge, background information, and domain-specific facts can enhance the model's understanding of the text and improve its performance on the task.

5. Data Sparsity and Domain Adaptation: Textual entailment models heavily rely on large annotated datasets. However, collecting large-scale entailment datasets across different domains and languages can be challenging. Models need to generalize well to new and unseen data, making domain adaptation and transfer learning crucial for their success.

In this report, we address the textual entailment task by proposing and implementing two transformer-based models. These models aim to overcome the challenges mentioned above by leveraging the power of attention mechanisms, contextual embeddings, and sequential processing. Our experimental findings and performance evaluation shed light on the effectiveness of these models in tackling the complexities of the textual entailment task.

2 Related work/Background

Three approaches have been proposed to solve the problem of TE: Semantic, syntactic, and lexical entailment recognition. They differ in their approaches to determining the relationship between two text fragments. Semantic entailment focuses on mapping language expressions to semantic representations, taking into account the meaning of the text. This approach can reveal similarities that are not visible at a syntactical or surface level, such as grammatical variations or different voices conveying the same meaning.

Syntactic entailment, on the other hand, considers the structural arrangement of words and phrases in the sentences. It looks at how the sentences are put together grammatically to determine the entailment relationship.

Lexical entailment recognition, in contrast, is based solely on lexical concepts. It works directly on the input string and performs string comparisons of the surface strings, focusing on word overlap, subsequence matching, and longest substring using a sliding window.

Recent works in recognizing textual entailment (RTE) have proposed innovative approaches to address this challenge. One such approach involves the use of deep learning models, specifically the Bidirectional Encoder Representations from Transformers (BERT) model developed by Google. This model has been widely utilized in the last three years to solve the task of RTE, demonstrating its effectiveness in natural language understanding (NLU) applications.

Another notable approach is the use of Attention Mechanisms to improve the performance of deep learning RTE models. Attention-based techniques have been employed to enhance accuracy in several primary studies. This technique continues to be developed to further increase the accuracy of RTE models, showcasing its potential in advancing the field of textual entailment recognition.

Furthermore, some researchers have focused on the enhancement of RTE datasets by including premise and hypothesis pairs that emphasize understanding the notion of entities and roles. This approach aims to provide more comprehensive and diverse data for training and developing deep learning-based RTE models.

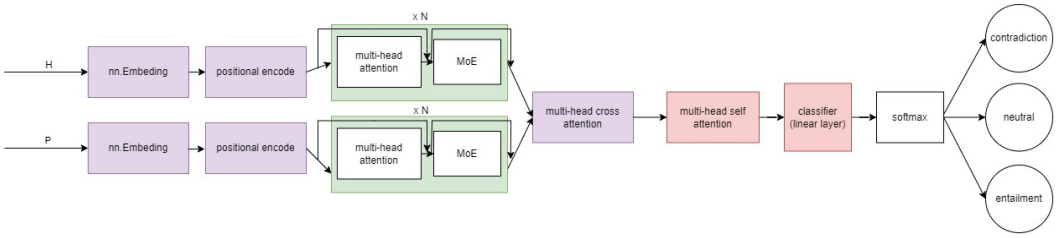

Figure 1: our first proposed model

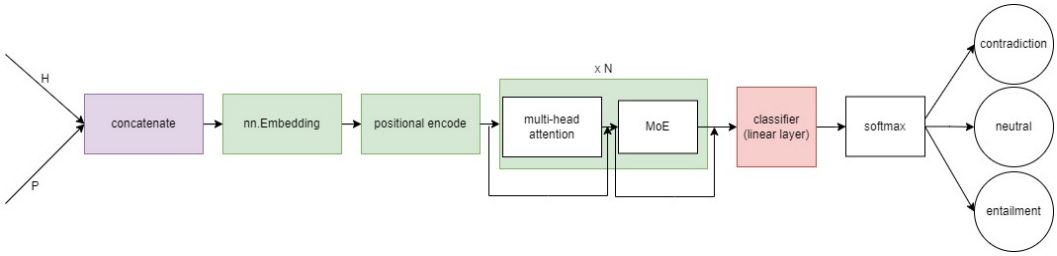

Figure 2: our second proposed model

These recent works demonstrate the ongoing exploration of advanced techniques and dataset improvements to address the challenges in recognizing textual entailment.

3 Proposed method

Our proposed method aims to improve the performance of textual entailment task by utilizing a transformer-based architecture with specific modifications. We have developed two models to address this problem, each with a distinct approach.

1. Model 1: Separate Embeddings: In the first model, we embed the hypothesis (H) and premise (P) separately. This means that each text fragment is independently encoded and processed. After the separate embeddings, the model predicts the label based on the individual context learned from each text fragment. Figure 1

2. Model 2: Concatenated Embeddings with Context Prediction: In the second model, we concatenate the hypothesis and premise embeddings and pass them through the transformer-based architecture. By considering the concatenated representation, the model captures the contextual relationship between H and P. The model then predicts the label based on the context computed in the [CLS] token, which summarizes the information from the concatenated embeddings. Figure 2

Both models aim to address the textual entailment task by leveraging transformer-based architectures. However, they differ in how they handle the input embeddings and utilize the contextual information. We provide comprehensive explanations of both models to present a clear understanding of our proposed method.

First model:

- H and P are embedded and positional encoded.
- We have Encoder Blocks which consist of a multi-head attention followed by the Mixture of Experts mechanism. The Mixture of Experts (MoE) neural network layer, introduced in 2024, combines the predictions of multiple specialized models to make a final prediction. As shown in Figure 3, it uses a gating network to determine the importance of each expert based on the input. Each expert focuses on a specific aspect of the data. The outputs of the experts are weighted and combined to produce the final prediction. The MoE layer is effective for handling complex and diverse datasets by leveraging the strengths of multiple models. It improves performance and generalization compared to using a single model. In the original form, each input vector is assigned to 2 of the 8 experts by a router. The layer's

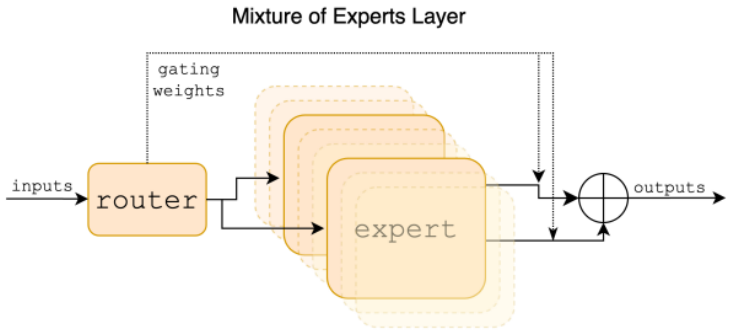

Figure 3: mixture of experts

output is the weighted sum of the outputs of the two selected experts. However, we decided to assign weights to all the experts and include all of their outputs in the result.

- H and P are separately fed to a sequence of Encoder blocks.
- The outputs are fed to a multi-head cross-attention algorithm to generate a single output.
- Subsequently, the output goes through a self-attention algorithm and finally, a linear layer classifies the output.
- We used a Softmax function at the final step. the result will be a 3-dimensional vector that has the probability of H and P belonging to each of the three classes.

Second model:

- P and H are first sequentially attached to each other. This concatenation allows the model to analyze the sentences bidirectionally, unlike our first model which analyzed only the presence of H in P. This means that we consider the presence of the information of P in H and H in P as a sign of a stronger entailment relationship.
- The output goes through embedding, positional encoding, and a sequence of Encoder Blocks.
- the result is classified by a linear layer and Softmax is applied as above.

4 Results

The SNLI (Stanford Natural Language Inference) dataset is a collection of approximately 570k sentence pairs that have been manually labeled by five annotators. Each annotator assigned a label (entailment, contradiction, or neutral) to each sentence pair, and the majority label was chosen as the final classification. This dataset was designed to aid in the development and evaluation of machine-learning models for sentence comprehension.

We performed a thorough EDA on this dataset and our findings are listed below.

Having 550152 rows and 14 columns, this dataset is the largest dataset for the task of TE. The columns are `gold_label`, `sentence1_binary_parse`, `sentence2_binary_parse`, `sentence1_parse`, `sentence1_parse`, `sentence1`, `sentence2`, `captionID`, `pairID`, `label1`, `label2`, `label3`, `label4`, `label5`. Out of these features, only `sentence1` (P), `sentence2` (H), and `gold_label` (class label) are useful for us. It has no duplicate values, but there are some values missing. Since only 6 rows of sentence 2 are missing, we simply drop them.

The unique values in the label column were 4 and their distribution is shown in Figure 4.

Therefore, we decided to drop all rows with the '-' label too. According to the database's documents, `gold_label` is the label chosen by the majority of annotators, but where no majority exists, this is '-', and the pair should not be included when evaluating hard classification accuracy. After dropping the rows of this class, the data becomes well-balanced, with each of the three remaining classes having approximately 180K rows. This makes our work easier since we will not need any resampling techniques to balance the data.

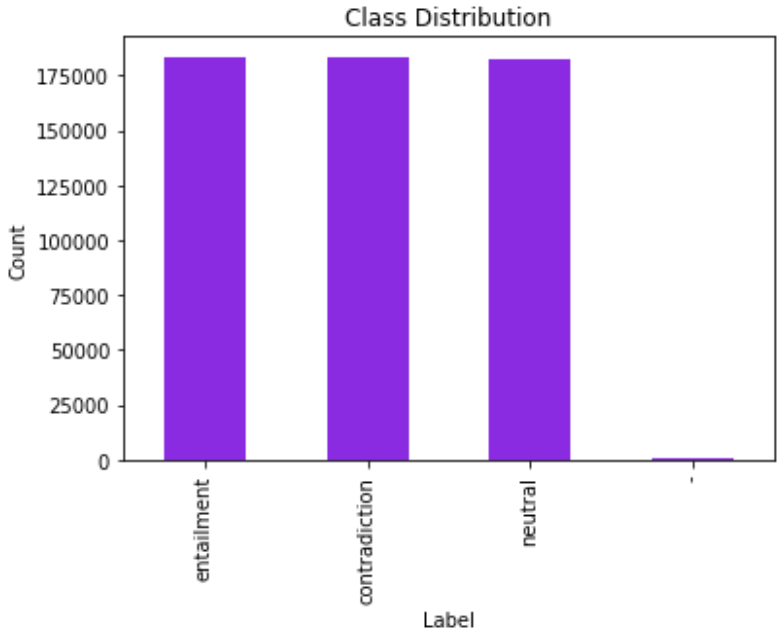

Figure 4: label distribution

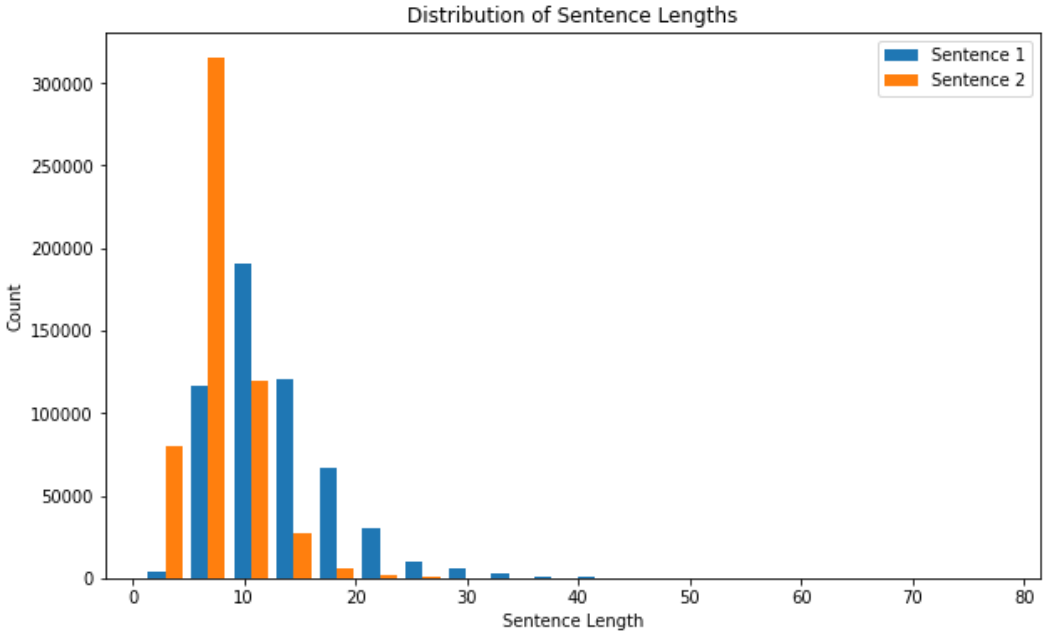

Figure 5: sentence length frequencies

For a simple look at the data's properties, we split the sentences based on the space delimiter and plotted their frequency in Figure 5.

We also plotted the most frequently used words in all sentences with a WordCloud in Figure 6

We also calculated the top 10 most common words in the premises and they were as follows. Figure 7 visualizes this matter.

- man: 151918

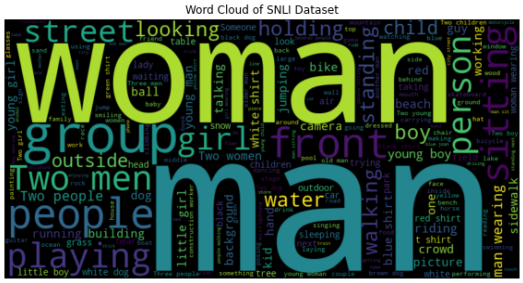

Figure 6: all sentences in corpus wordcloud

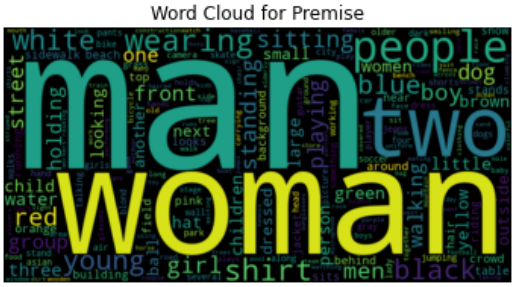

Figure 7: most frequently used words in Premises

- woman: 79007
- two: 75748
- people: 60205
- wearing: 60205
- shirt: 57242
- white: 50465
- black: 47326
- young: 46762
- blue: 43075

And for hypothesis:

- man: 114462
- people: 60960
- woman: 58705
- two: 46527
- playing: 29429
- men: 27690
- girl: 25470
- boy: 24996
- dog: 23576
- outside: 22350

We also did an analysis of the n-grams of the Premises. This is the list of the top 10 most common (2, 2)-grams:

- two men: 14812

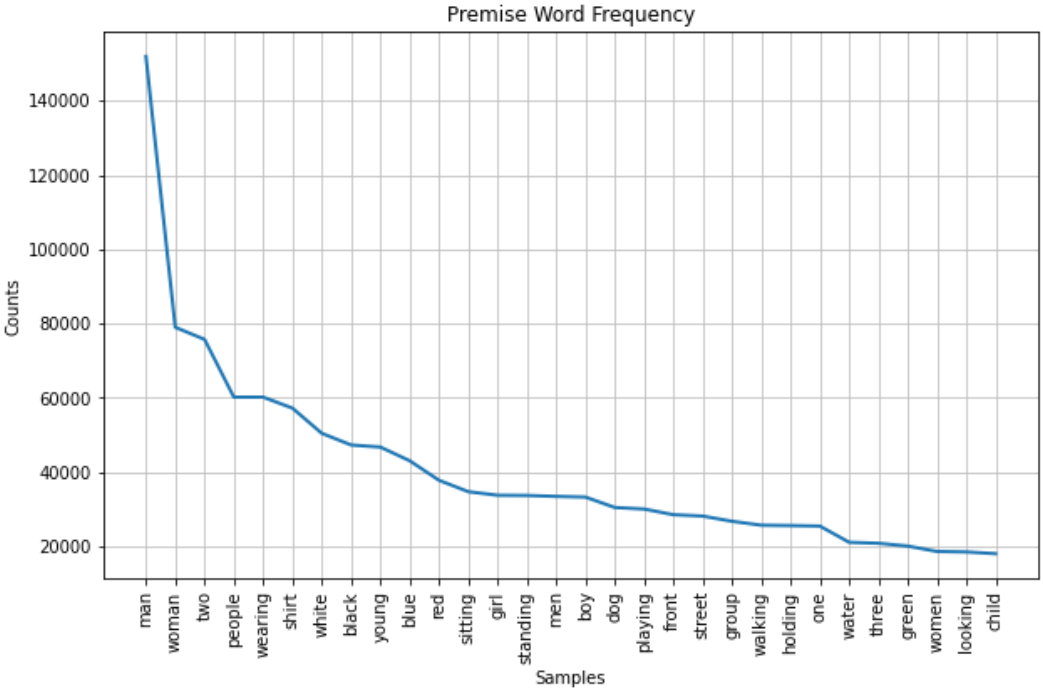

Figure 8: premise word frequency

- man wearing: 13762
- street .: 12080
- group people: 11047
- water .: 9815
- background .: 9240
- , one: 8986
- young boy: 8316
- two people: 7996
- blue shirt: 7643

The analysis for Part of Speech tags results are also demonstrated in Figure 10 and Figure 11.

The provided plots illustrate the distribution of different parts of speech in the analyzed text. To understand the plots, it is necessary to familiarize ourselves with the abbreviations used for the Part-of-Speech (POS) tags. These tags represent various linguistic categories, such as nouns, verbs, adjectives, and more. Let's examine some examples to gain a better understanding:

- NN: Noun (singular or mass) - Examples: "cat," "book," "team"
- DT: Determiner - Examples: "the," "a," "an"
- IN: Preposition or subordinating conjunction - Examples: "in," "on," "at," "with"
- VBG: Verb, gerund or present participle - Examples: "running," "swimming," "playing"
- JJ: Adjective - Examples: "big," "happy," "red"
- VB: Verb, base form - Examples: "run," "eat," "jump"
- RB: Adverb - Examples: "quickly," "very," "happily"
- PRP: Personal pronoun - Examples: "I," "you," "he," "she"
- CC: Coordinating conjunction - Examples: "and," "but," "or"

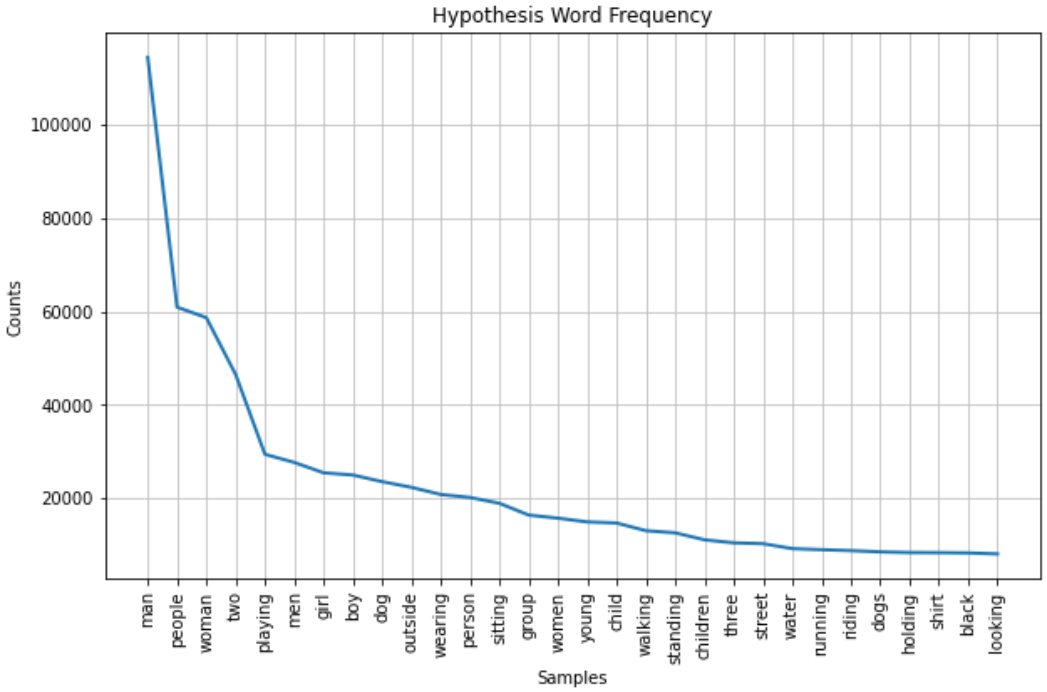

Figure 9: hypothesis word frequency

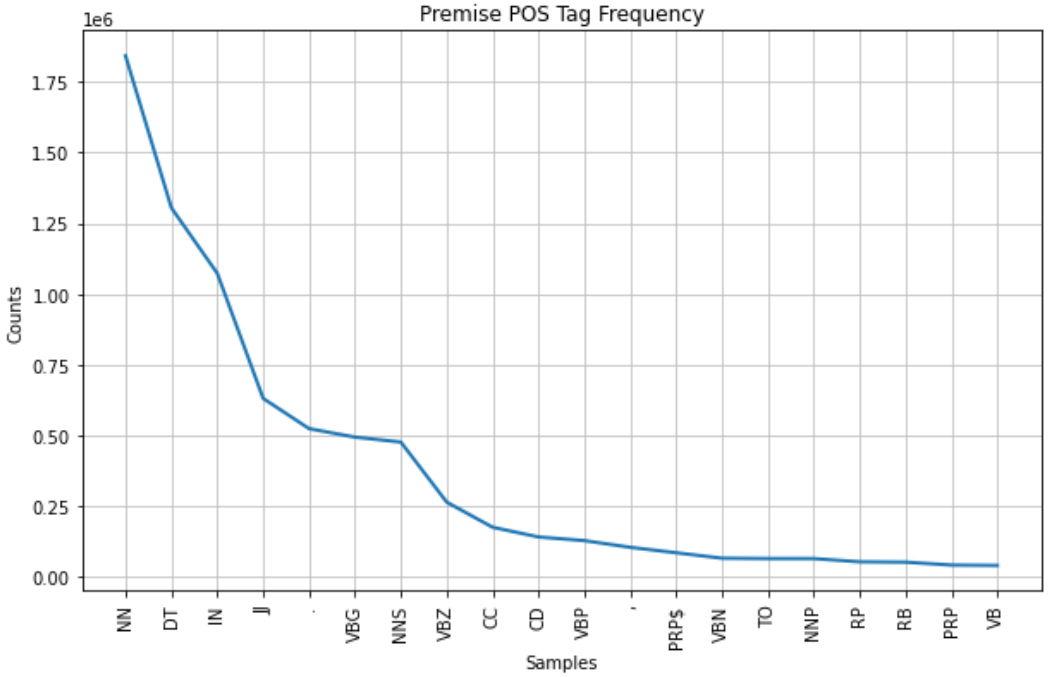

Figure 10: premise pos tag frequency

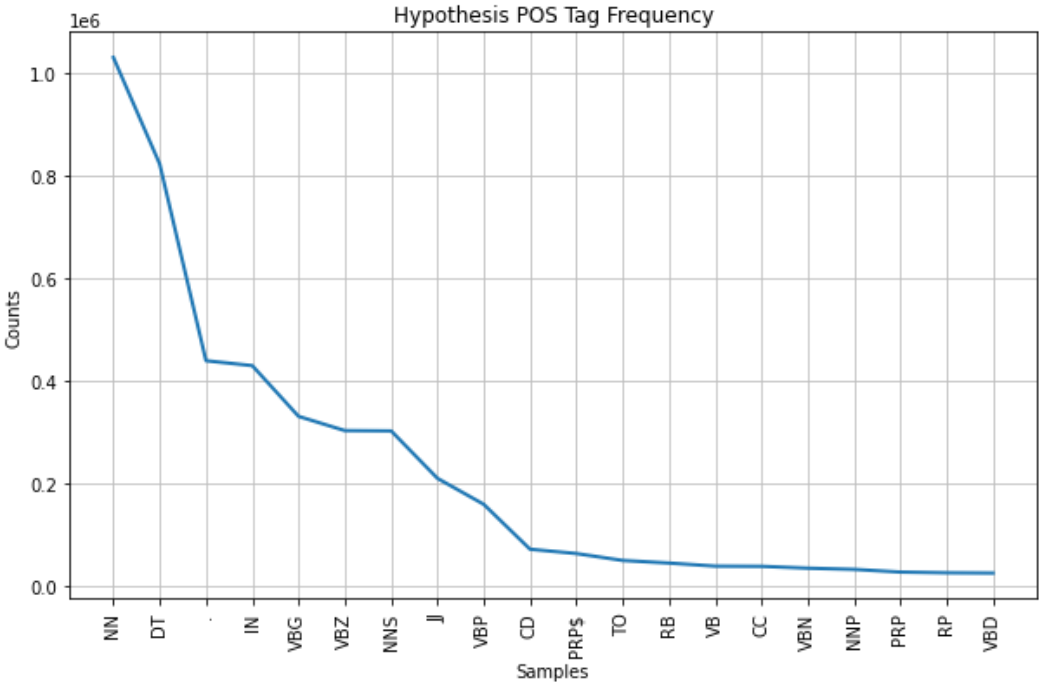

Figure 11: hypothesis pos tag frequency

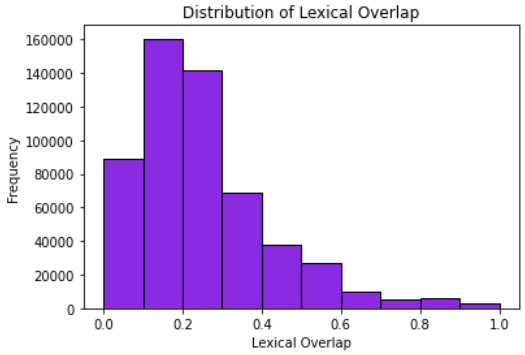

Figure 12: lexical overlap disribution

- CD: Cardinal number - Examples: "one," "two," "three"

By analyzing the distribution of these POS tags, we can draw important insights and implications from the plot. For example, a high frequency of VBG might indicate a text that emphasizes ongoing activities, while a significant presence of NN suggests a focus on objects or entities. These patterns provide valuable information about the linguistic characteristics and style of the analyzed text.

Another interesting measure that we could extract was lexical overlap. This is measured by the size of the intersection of H and P divided by the size of their union. This provides information about the degree of similarity or shared vocabulary between different texts or document pairs. The Figure 12 shows this measurement's distribution.

The average lexical overlap between the pairs is 0.247557, indicating a moderate level of similarity in terms of shared vocabulary. The standard deviation is 0.170182, representing the variability or spread of the data around the mean. The minimum lexical overlap value is 0, indicating that there are pairs of texts with no common words, and the maximum lexical overlap value is 1, signifying that there are pairs of texts with a complete or exact lexical match.

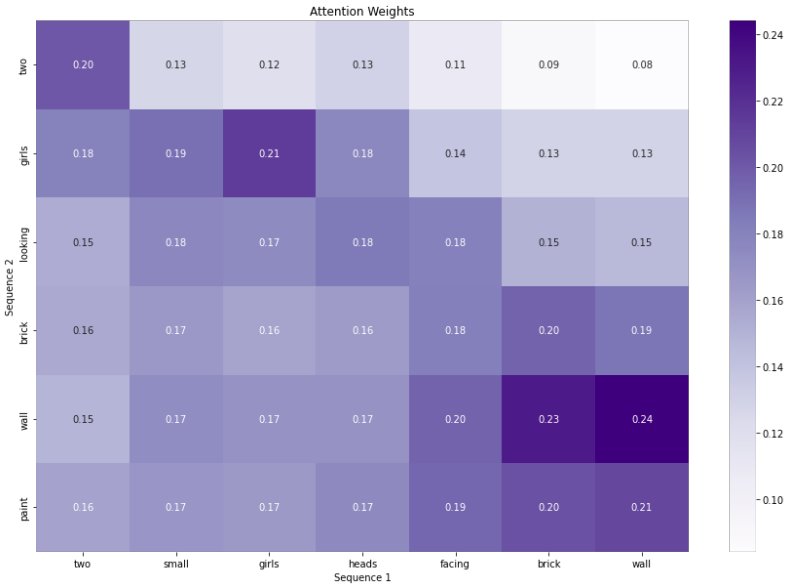

Figure 13: heatmap of cross-attention for a neutral pair

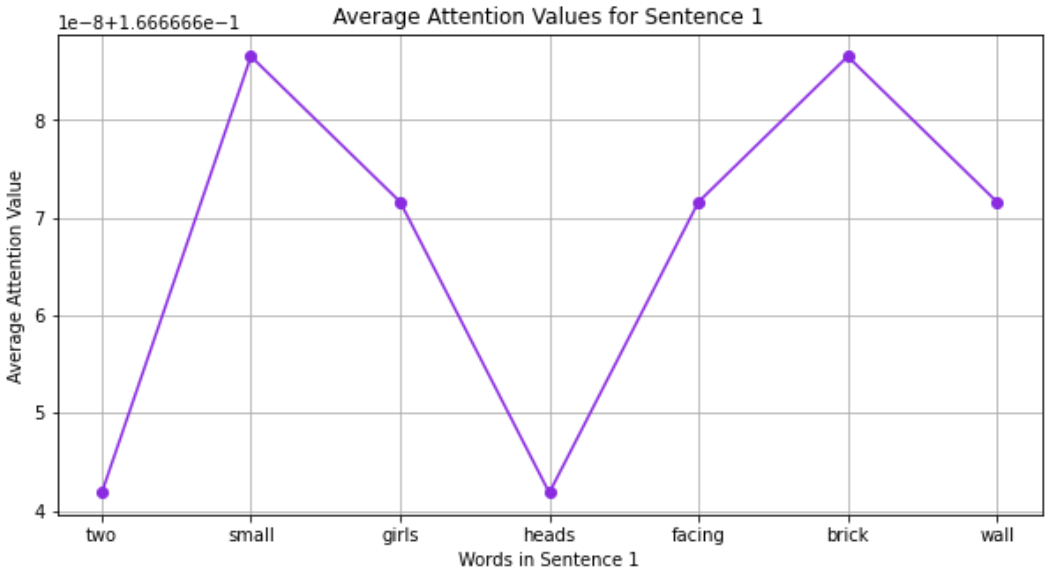

Figure 14: line plot of cross-attention for a neutral pair

Finally, we implemented the cross-attention calculation matrix and tried to analyze some random rows of the dataframe and visualized the cross-attention between H and P. See Figure 13, Figure 14, Figure 15, Figure 16, Figure 17, Figure 18, and Figure 19, Figure 20, Figure 21. Notice that we have first tokenized the sentences, made them all lowercase, removed unnecessary words, removed stopwords, and then calculated and visualized attention matrices.

We also used BertViz's built-in library to visualize attention weights for another random pair in Figure 22.

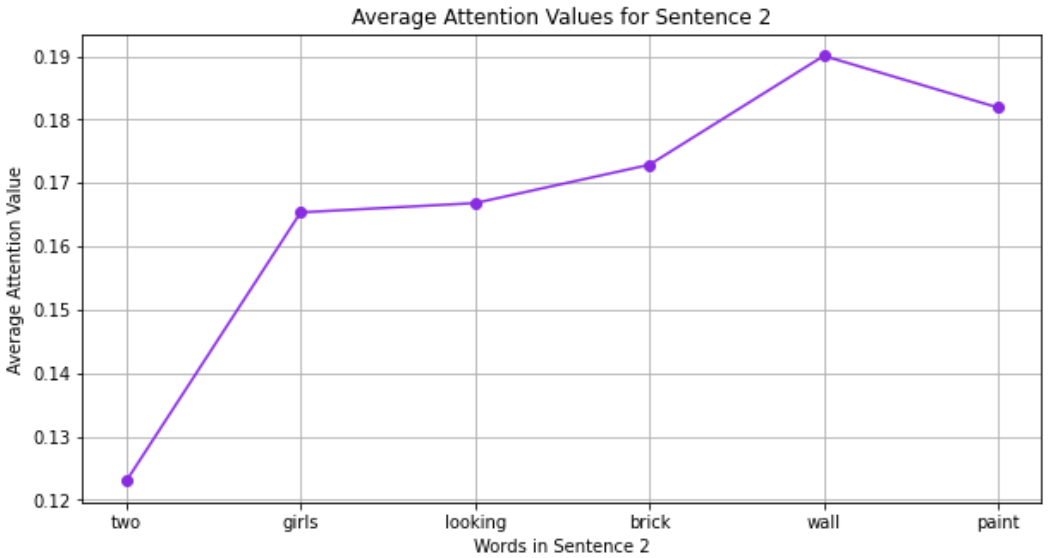

Figure 15: line plot of cross-attention for a neutral pair

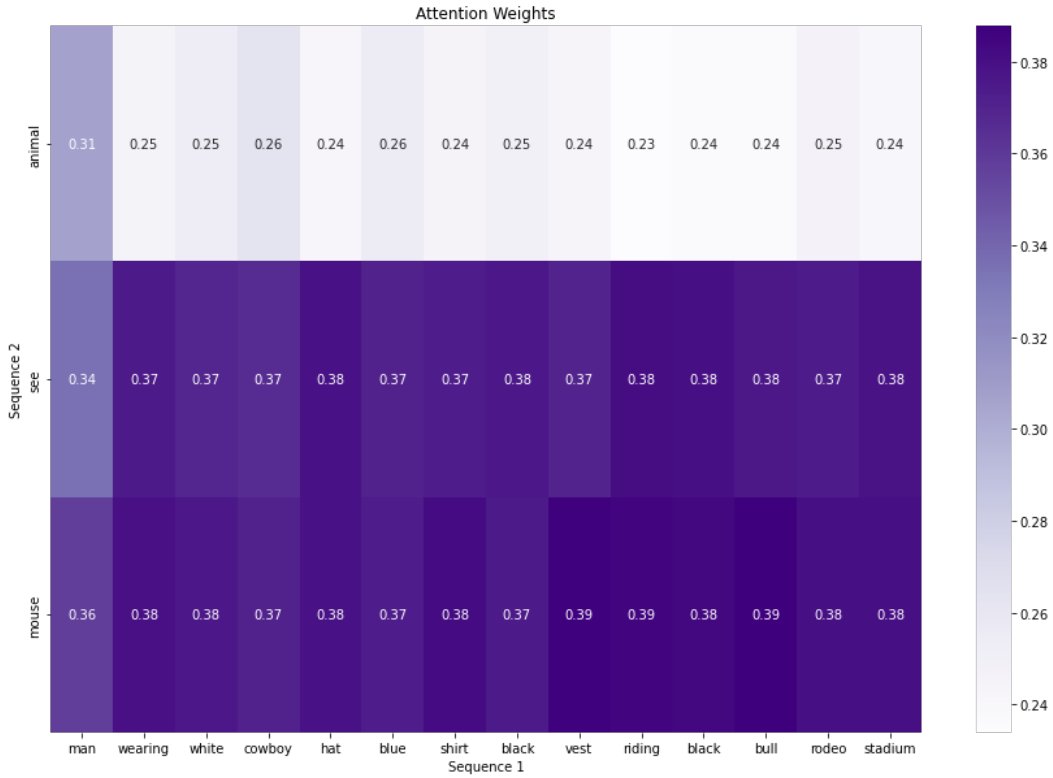

Figure 16: heatmap of cross-attention for a contradictory pair

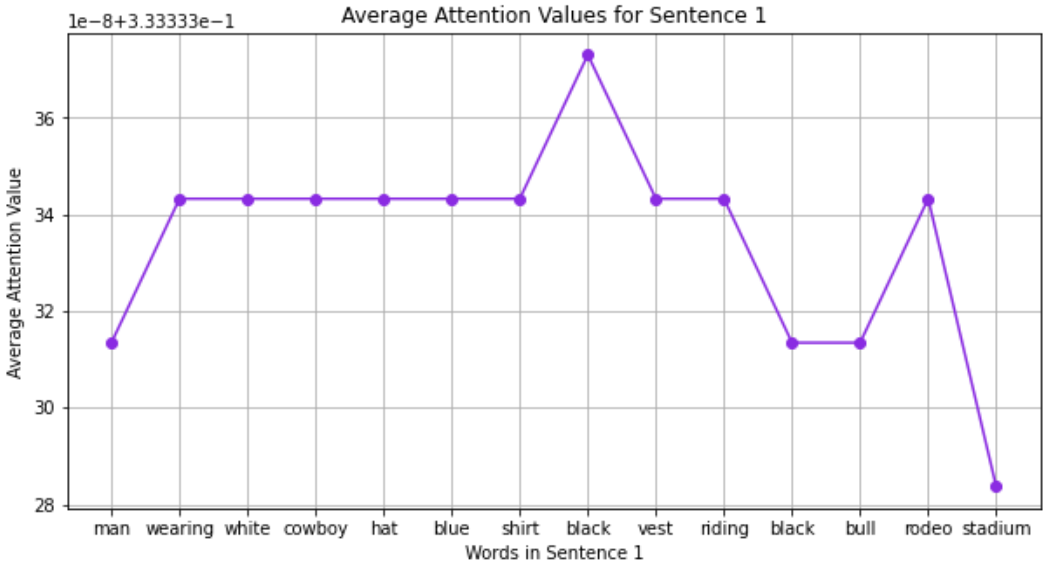

Figure 17: line plot of cross-attention for a contradictory pair

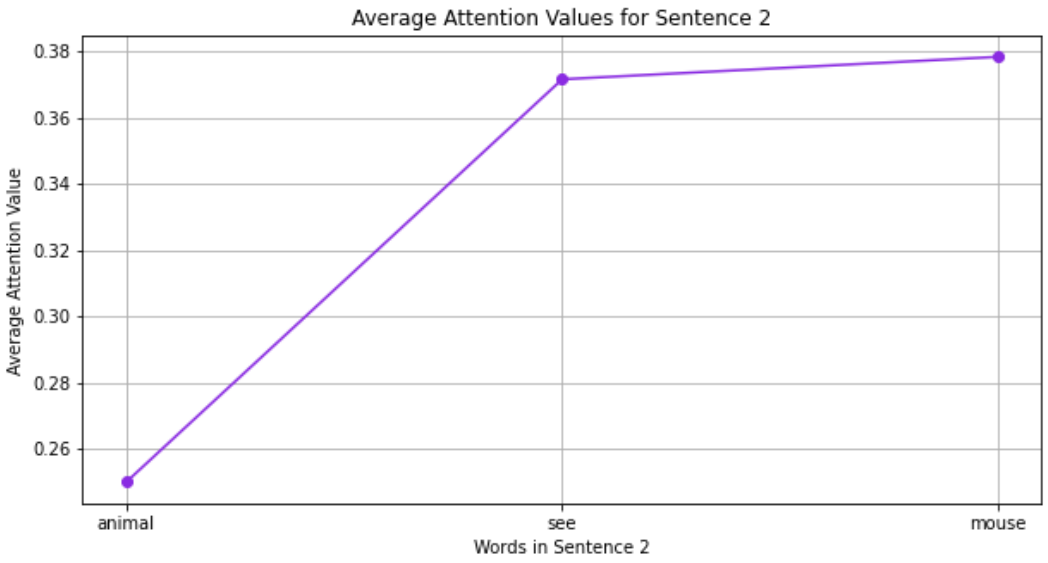

Figure 18: line plot of cross-attention for a contradictory pair

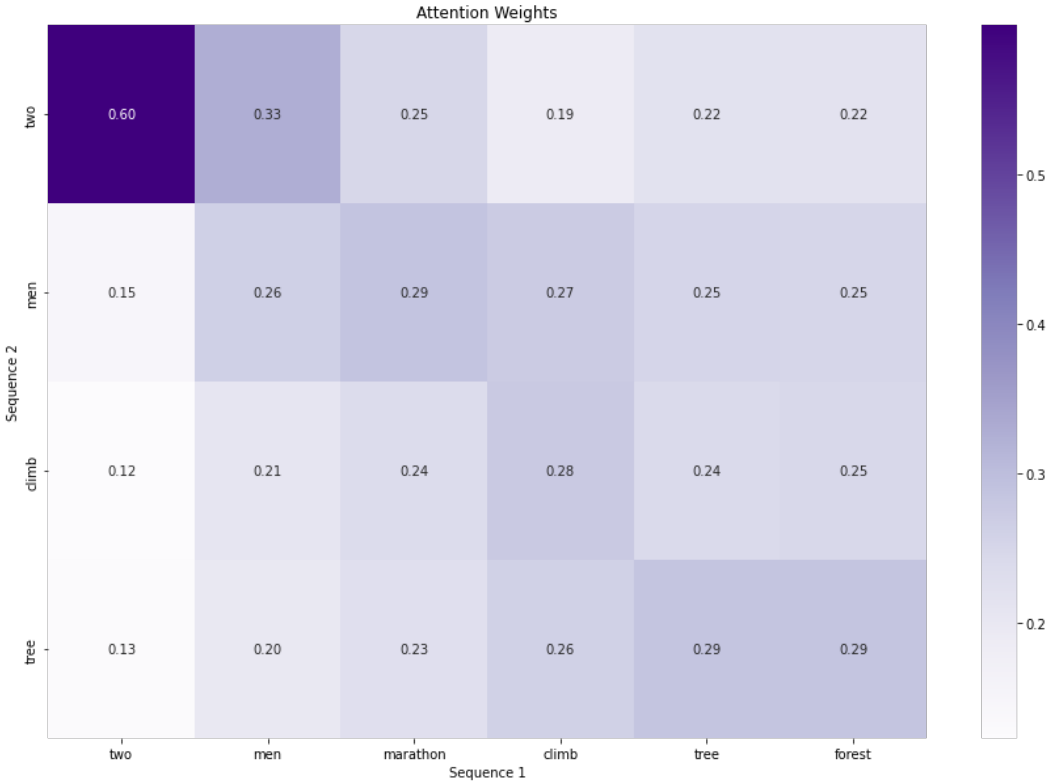

Figure 19: heatmap of cross-attention for an entailment pair

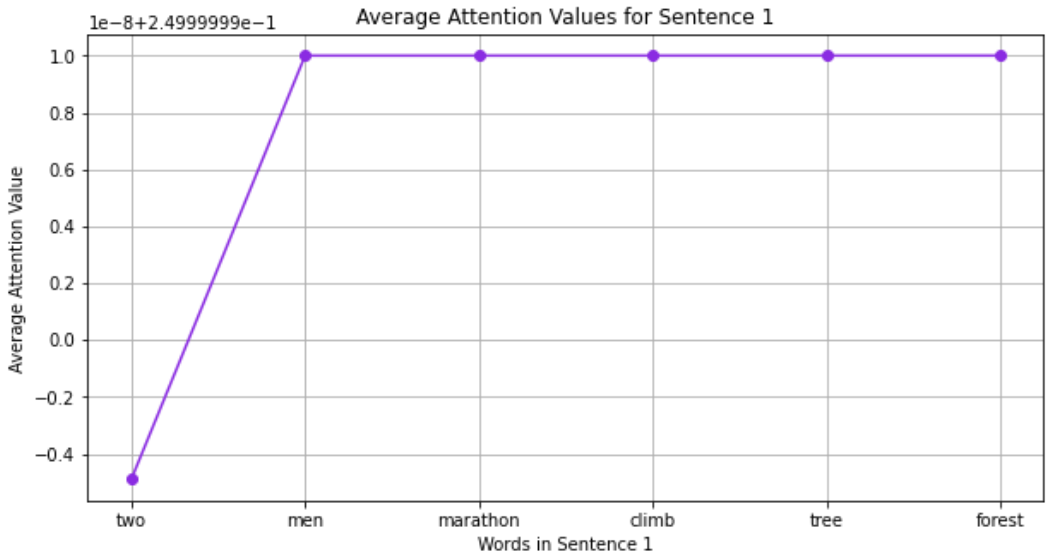

Figure 20: line plot of cross-attention for an entailment pair

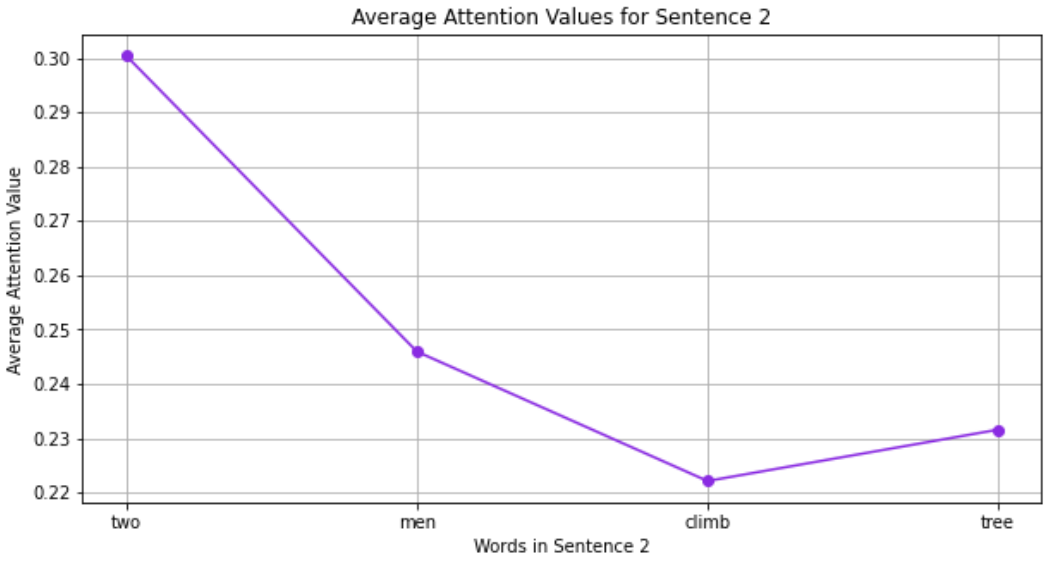

Figure 21: line plot of cross-attention for an entailment pair

Figure 22: cross-attention visualization for an entailment pair

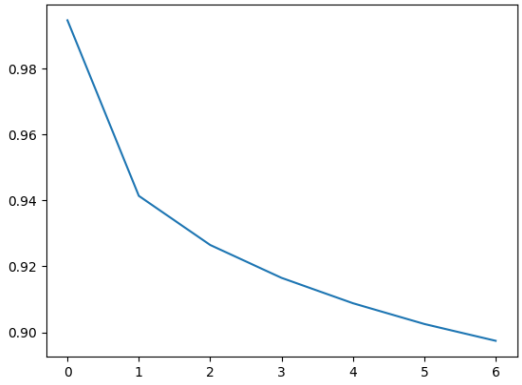

Figure 23: losses for the first model

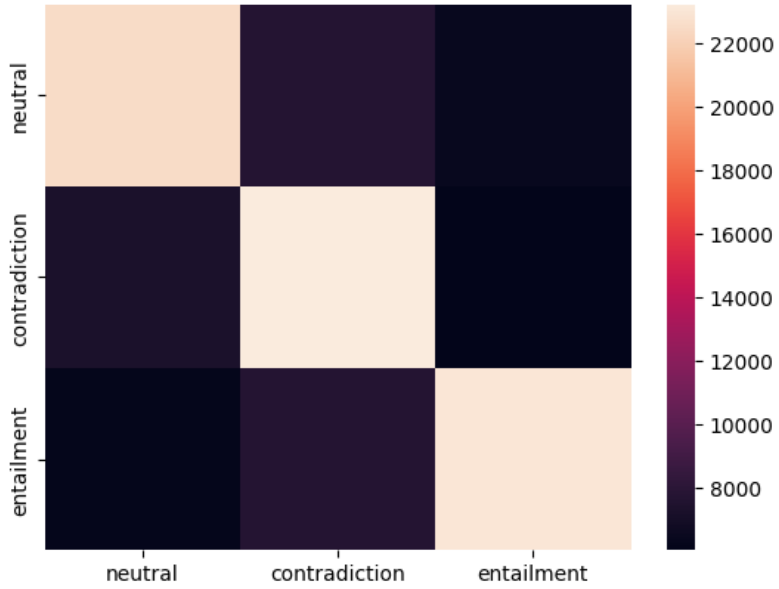

Figure 24: confusion matrix for the first model

5 Discussion

We first split our data into training and validation sets with a ratio of 80:20. We trained both our models on the training dataset with 7 epochs. The results were tested using the validation set. Eventually, the performance of our models was measured via 4 metrics: accuracy, F1 score, AUC score, and confusion matrix. As seen in Figure 23 training losses for the first model decrease well over epochs. Furthermore, Figure 24 shows how well the model performed. The metrics are reported as follows:

- Accuracy: 0.6258646424930828
- AUC: 0.719410120833868
- F1 Score: 0.6259852473373025

As for model 2, the evaluated metrics were:

- Accuracy: 0.6005806756953546
- AUC: 0.7003569370801249

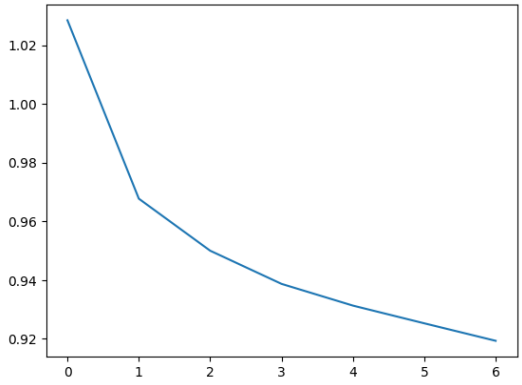

Figure 25: losses for the second model

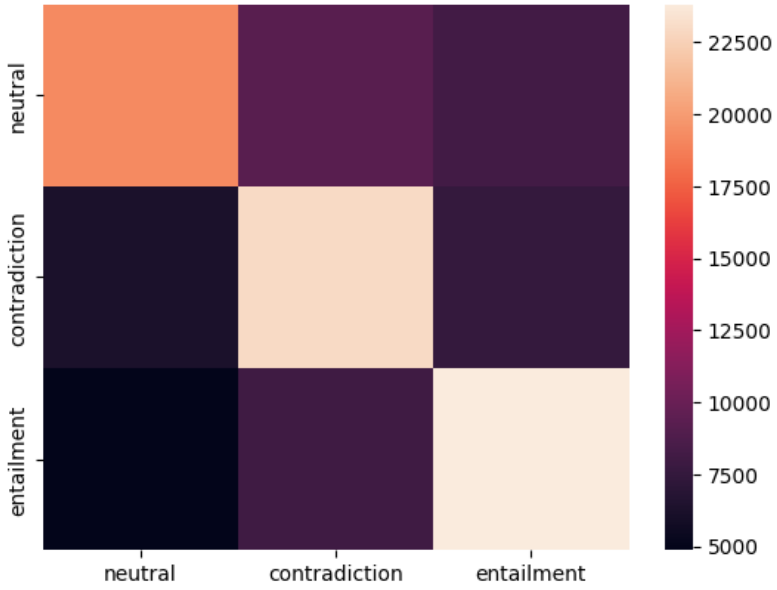

Figure 26: confusion matrix for the second model

- F1 Score: 0.5995687113675093

And Figure 25 shows the decrease in losses, Figure 26 demonstrates this model's performance with its confusion matrix.

We can conclude that the performance of model 2, which was a simpler version of the first one was poorer. In both models, the decrease in the losses and the good scores despite a low number of epochs show that our architectures were reasonable choices. However, with the increase in the complexity and layers of the network, the model can perform more accurately.

The specific type of deep neural networks we have implemented, which is based on transformer models, is suitable for the textual entailment task for several reasons:

1. Transformer models have shown excellent performance in natural language processing tasks, including textual entailment. They are known for their ability to capture long-range dependencies and contextual information effectively.

2. The multi-head attention mechanism in our encoder block allows the model to attend to different parts of the input sequence simultaneously, capturing different aspects of the relationship between

the premise (P) and the hypothesis (H). This attention mechanism helps the model focus on relevant information and capture the interactions between the two text sequences.

3. The positional encoding step in both models ensures that the model takes into account the order of words in the input sequences. This is crucial in understanding the semantic relationships between words and their positions, which is important for textual entailment.

4. The Mixture of Experts component can improve the model's performance by combining the strengths of different experts or sub-models. This allows the model to capture various aspects of the data, understand each token's complexity fully, and make more accurate predictions.

The benefits of the proposed model over other approaches can include:

1. Enhanced representation learning: The multi-head cross-attention mechanism allows the model to capture intricate relationships between the premise and hypothesis. By attending to different parts of the input, the model can learn more robust representations of the text, which can lead to better performance in understanding TE.

2. Contextual information modeling: The self-attention mechanism in both models enables the model to capture contextual information from the entire input sequence. This helps in understanding the dependencies and interactions between words.

3. Flexibility and adaptability: The transformer-based models are highly flexible and can be easily adapted to different tasks and datasets. By utilizing the transformer architecture, our models can provide a strong foundation for TE.

4. Efficient training and inference: Transformer models can be trained efficiently in parallel on modern hardware. The self-attention mechanism allows for parallel computation, which speeds up training and inference, making the models practical and scalable.

References

[1] Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M., Stock, P., Subramanian, S., Yang, S., . . . Sayed, W. E. (2024). Mixtral of Experts. ArXiv. /abs/2401.04088

[2] Devlin, J., et al. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv preprint arXiv:1810.04805.

[3] Dagan, I., Glickman, O., & Magnini, B. (2006). Recognizing Textual Entailment: Models and Applications. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[4] Bos, J., Markert, K., & Zhou, Z. (2006). Syntactic and Semantic Analysis for Recognizing Textual Entailment. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (ACL)*.

[5] Mohammad, S. M., & Turney, P. D. (2013). A Lexical-Semantic Approach to Sentiment Detection in Twitter Messages. *Journal of Artificial Intelligence Research, 47*, 1-50.