IUST-NLPLAB at SemEval-2024 Task 9: BRAINTEASER By MPNet (Sentence Puzzle)

Mohammad Hossein Abbaspour, Erfan Moosavi Monazzah, Sauleh Eetemadi
Iran University of Science and Technology
m_abbaspoor80, moosavi_m@comp.iust.ac.ir
sauleh@iust.ac.ir

Abstract

This study addresses a task encompassing two distinct subtasks: Sentence-puzzle and Word-puzzle. Our primary focus lies within the Sentence-puzzle subtask, which involves discerning the correct answer from a set of three options for a given riddle constructed from sentence fragments. We propose four distinct methodologies tailored to address this subtask effectively. Firstly, we introduce a zero-shot approach leveraging the capabilities of the GPT-3.5 model. Additionally, we present three fine-tuning methodologies utilizing MPNet as the underlying architecture, each employing a different loss function. We conduct comprehensive evaluations of these methodologies on the designated task dataset and meticulously document the obtained results. Furthermore, we conduct an in-depth analysis to ascertain the respective strengths and weaknesses of each method. Through this analysis, we aim to provide valuable insights into the challenges inherent to this task domain.

1 Introduction

The remarkable efficacy of language models in navigating complex reasoning tasks, particularly in the realm of vertical thinking, has prompted their exploration in lateral thinking problem domains (Waks, 1997). One such domain, exemplified by the BRAINTEASER task (Jiang et al., 2024), entails a multiple-choice Question Answering framework comprising two distinct subtasks: Sentence-puzzle and Word-puzzle. This paper directs its focus toward the Sentence-puzzle subtask, which hinges on unraveling the intricate nuances of common sense embedded within sentence fragments (Jiang et al., 2023).

Initially, we adopted a zero-shot approach, followed by experimentation with three distinct fine-tuning methodologies tailored for Language Model (LLM) architectures as the backbone. Additionally, we have made our code openly accessible on GitHub[1] to facilitate reproducibility and further research endeavors.

A primary challenge we encountered pertained to the constraint imposed by the dataset size, posing impediments to both fine-tuning procedures and model training from scratch. To mitigate this challenge, we employed various strategies, including the utilization of k-fold cross-validation techniques, to enhance the robustness and generalizability of our approach.

2 Background

In the implementation of the zero-shot method, we employed ChatGPT-3.5, utilizing a consistent prompt template throughout. Conversely, for the fine-tuning process, we adopted a pre-trained sentence embedding technique to map input sentences into meaningful numerical vectors, facilitating subsequent decision-making regarding the provided questions and options.

Furthermore, in our fine-tuning methodologies, we integrated two distinct types of loss functions: Binary Cross-Entropy loss and Triplet loss. The Binary Cross-Entropy loss function operates on the premise of determining whether two sentences coherently match or not. Conversely, the Triplet loss function aims to optimize the proximity between the question and the correct answer while concurrently ensuring a clear distinction between the question and unrelated options.

3 Dataset

The task dataset comprises 507 samples designated for training purposes, with an additional 120 samples allocated for the test set. Notably, the evaluation set encompasses a subset of the training samples, necessitated by data scarcity. Consequently, for two out of the three fine-tuning methods, no

[1] https://github.com/MohammadHAbbaspour/SemEval-2024_task9_BRAINTEASER

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 1106–1109
June 20-21, 2024 ©2024 Association for Computational Linguistics

data from the training set were utilized for evaluation. Conversely, the third method, employing the k-fold technique, leveraged the training samples for both the training and evaluation phases.

4 System overview

4.1 Zero Shot

For the zero-shot methodology, we leveraged the ***GPT-3.5-turbo*** model, utilizing a temperature parameter of 0.0. To elicit responses from the model, we employed a consistent prompt template outlined in Table 1. Within this template, we systematically substituted the question and available options with the corresponding tokens. Additionally, we extracted explanations from the model to facilitate a deeper analysis of its reasoning processes.

4.2 Binary Classification

In this approach, we utilized the ***all-mpnet-base-v2*** model (Song et al., 2020; Jayanthi et al., 2021) as the backbone, which was subsequently frozen. Following this, we introduced a trainable layer for inference purposes. The core principle underlying this method involves the transformation of each sample within the dataset, comprising a question and three options (excluding the 'None of the above' option), into three distinct pairs. Each pair encompasses the question alongside one of its options, with a corresponding label indicating whether the option constitutes the correct answer. Consequently, the original training dataset, comprising 507 samples, was expanded to form a new dataset comprising 1521 samples.

Moreover, during the process of feeding sentences into the model, we initially present the question and option to the backbone model. Subsequently, we concatenate the resulting vectors and forward them to the inference layer. For the final decision-making step, we apply a sigmoid function to the output of the inference layer, enabling us to ascertain the consistency between the two sentences by employing a threshold of 0.5.

During the inference stage, we determine the option with the highest score among the three available options.

4.3 Triplet loss

In this approach, our base model and backbone remain consistent with the previous section. However, the data preparation process differs. In the original dataset, each sample consists of a single question alongside three options, one of which is designated as the correct answer. Consequently, for each sample in the original dataset, we generate two samples in the new dataset. As a result, the new dataset comprises 1014 samples, with each sample comprising a question as the anchor, the correct answer as the positive, and a distractor as the negative.

As previously elucidated, the fundamental concept is to minimize the distance between the question and the correct answer while maximizing the distance between the question and unrelated options. To achieve this objective, we integrate a pre-trained sentence embedding model within the inference component.

Within our implementation, the inference component consists of two subparts: one dedicated to the anchor and the other to the positive and negative instances. The anchor component essentially functions as an identity layer, as it cannot glean meaningful insights from a single question. Conversely, the other component aims to discern the disparities between the positive and negative instances by leveraging information from the question. Hence, we concatenate the output of the sentence embedding model for the question and the correct answer to form the positive instance, and likewise for the question and the distractor to constitute the negative instance within the triplet loss framework (see Algorithm 1).

Algorithm 1 Algorithm of the triplet loss

```
procedure FORWARD($qemb, ansemb, disemb$)
    anchor = $qemb$
    positive = concatenate($qemb, ansemb$)
    negative = concatenate($qemb, disemb$)

    anchor = anchor_inference(anchor)
    positive = option_inference(positive)
    negative = option_inference(negative)
return anchor, positive, negative
```

4.4 Triplet loss (K-Fold)

As previously highlighted, the limited availability of data poses a significant challenge in this task. To address this issue, we adopt the K-Fold technique, a commonly employed strategy for mitigating data scarcity. The key components of this approach remain consistent with the previous sections, including the backbone model and the underlying algorithm. However, the distinguishing factor lies

Prompt

```
Which option is the answer to this riddle, explain in a step-by-step manner:
<RIDDLE>
1) <OPTION1>
2) <OPTION2>
3) <OPTION3>
4) None of the above.
 Please place your answer in a JSON format:
{
  "option_number": <JUST_THE_NUMBER_OF_THE_CORRECT_OPTION>,
  "explanation": <EXPLANATION_WHY_IT_IS_CORRECT>
}
```

Table 1: Constant template for prompt used in zero-shot

in the training process, wherein multiple models are trained, and the most performant one is selected as the final iteration based on its performance on the evaluation data.

Initially, we partition the training dataset into **k** folds, each serving as the basis for training a distinct model utilizing the Triplet loss. Subsequently, a subset of validation data is extracted from each fold, and the model is trained on the remaining data. Evaluation of each model is then conducted on the evaluation data corresponding to its respective fold. Upon completion of the training process, **k** models are obtained. To select the final model, we employ a sorting criterion based on the following key metric:

$$key = \frac{val\_acc}{train\_loss}$$

This metric encapsulates the trade-off between validation accuracy and training loss. The selection process involves sorting the models based on this key metric and choosing the middle model. This decision is predicated on the objective of maximizing validation accuracy while minimizing training loss. However, it is important to note that models with the highest values of **key** may exhibit signs of overfitting and possess reduced generalization capabilities. Hence, opting for the middle model mitigates the risk of overfitting and ensures enhanced generalization.

5 Experimental setup

5.1 Customized triplet loss

In methodologies utilizing the triplet loss paradigm, the loss function undergoes customization. While the original triplet loss hinges on the calculation of the **Euclidean distance** as a measure of difference, our approach diverges by customizing this metric to **cosine similarity** (see Algorithm 2).

Algorithm 2 Triplet loss, customized by cosine similarity

```
procedure LOSS($anchor, positive, negative$)
    positive_sim = cosine_similarity($anchor$, $positive$)
    negative_sim = cosine_similarity($anchor$, $negative$)
    loss = $negative\_sim$ - $positive\_sim$ + $margin$
    return loss
```

The maximum value of cosine similarity between two vectors is 1 which means two vectors are the same, and the minimum value between them is -1 which means they are different. So:

$$-2 \leq positive\_sim - negative\_sim \leq 2$$

We add the **margin=2** value to the loss for shifting it in positive numbers:

$$0 \leq positive\_sim - negative\_sim + margin \leq 4$$

Hence, if two vectors are the same it means the loss is equal to 0 and if two vectors are opposite it means the loss has its max value.

5.2 Hyperparameters

For the training of the discussed models, we scrutinized the hyperparameters outlined in Table 2.

6 Results

We conducted evaluations of the aforementioned methods on the test set, and the results are presented in Table 3. It is evident that the zero-shot method exhibits the best performance on the

	epochs	learning rate	batch size	validation size	k
Binary classification	10	0.001	4	-	-
Triplet loss	10	0.001	16	-	-
K-Fold	10	0.001	16	20	3

Table 2: Hyperparameters of models while training

	S_ori	S_sem	S_con	S_ori_sem	S_ori_sem_con	S_overall
Zero-shot	**0.7**	0.575	0.575	0.55	0.35	0.61
Binary classification	0.625	0.625	0.525	0.625	0.475	0.5916
Triplet loss(modified system)	0.65	**0.65**	**0.625**	**0.65**	**0.525**	**0.641**
K-Fold	0.6	0.6	**0.625**	0.6	0.5	0.608

Table 3: Comparison of our results

Original sentences. However, its efficacy diminishes notably when applied to other categories such as **Semantic**, showcasing a significant disparity compared to its performance on the **Original sentences**. Conversely, all of our fine-tuning methods demonstrate comparable performance across all categories.

Among our fine-tuning methodologies, the **Triplet loss** approach stands out with the most impressive performance, achieving the highest **Overall** score among all methods.

The uniformity in scores observed with the **Triplet loss** method suggests that it does not exhibit bias towards specific words or segments of the sentence; rather, it considers the entire sentence holistically. This is in contrast to the zero-shot method, where significant discrepancies exist among its scores. However, it's worth noting that the performance of our fine-tuning models could potentially improve with a larger volume of data.

7 Conclusion

In this paper, we have presented four distinct methodologies for the BRAINTEASER task, a novel challenge involving common sense reasoning and sentence puzzle solving. We have evaluated our methods on the task dataset and compared their performance across different categories. Our results show that the zero-shot approach, based on GPT-3.5-turbo, achieves the highest score on the original sentences, but fails to generalize well to other categories. On the other hand, our fine-tuning methods, based on MPNet and various loss functions, demonstrate more consistent and robust performance across all categories, with the triplet loss approach achieving the best overall score. We have also employed the K-Fold technique to mitigate the data scarcity issue and enhance the generalization capability of our models. Through our analysis, we have provided valuable insights into the strengths and weaknesses of each method, as well as the challenges inherent to this task domain. We hope that our work will inspire further research on this novel and intriguing problem of common sense reasoning and sentence puzzle solving.

References

Sai Muralidhar Jayanthi, Varsha Embar, and Karthik Raghunathan. 2021. Evaluating pretrained transformer models for entity linking in task-oriented dialog.

Yifan Jiang, Filip Ilievski, and Kaixin Ma. 2024. Semeval-2024 task 9: Brainteaser: A novel task defying common sense. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1996–2010, Mexico City, Mexico. Association for Computational Linguistics.

Yifan Jiang, Filip Ilievski, Kaixin Ma, and Zhivar Sourati. 2023. BRAINTEASER: Lateral thinking puzzles for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14317–14332, Singapore. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding.

Shlomo Waks. 1997. Lateral thinking and technology education. *Journal of Science Education and Technology*, 6(4):245–255.