1. Team Name: Project Group 2

2. Team Members: Hung Nguyen, Autri Ilesh Banerjee, Jawad Tawsik

## 3. Task and Data

For this project, we are tackling the *SemEval 2024 BRAINTEASER* task, which involves training models to solve lateral thinking puzzles. We use two datasets, SP-train and WP-train, designed for Sentence Puzzles (SP) and Word Puzzles (WP), respectively. These datasets contain questions, correct answers, and distractors, formatted to challenge models with creative reasoning and problem-solving skills.

Each question in the dataset has adversarial versions of the original data in two ways - **Semantic Reconstruction**, wherein original questions are rephrased semantically without changing answers or distractors; **Context Reconstruction wherein t**he original reasoning routes are preserved, but a new situational setting is presented. ​​

The datasets are split into training, validation, and test sets, with SP-train containing 405 questions and WP-train containing 315 questions. Our goal is to develop models that can accurately identify the correct answer from a list of options, even when questions are rephrased or placed in new contexts. To simplify the testing, we have combined each pair of train, validation, and test datasets into one dataset each.

## 4. Resources List:

Models: [RoBERTaV2 Model](https://huggingface.co/docs/transformers/en/model_doc/roberta), [Llama 3.2 3B Model](https://huggingface.co/meta-llama/Llama-3.2-3B), [Phi-3.5 Mini Model](https://huggingface.co/microsoft/Phi-3.5-mini-instruct), [DeBERTaV2 Model](https://huggingface.co/docs/transformers/en/model_doc/deberta-v2), [Mixtral-8x7B Model](https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1), [MPNet-base-v2 Model](https://huggingface.co/sentence-transformers/all-mpnet-base-v2)

Libraries: [PyTorch](https://download.pytorch.org/whl/cu118), [Numpy](https://numpy.org/), [Transformers](https://huggingface.co/docs/transformers/en/index), [Protobuf](https://github.com/protocolbuffers/protobuf/tree/main/python#installation), [Tiktoken](https://github.com/openai/tiktoken), [Sentencepiece](https://github.com/google/sentencepiece), [Pandas](https://pandas.pydata.org/), [Scipy](https://docs.scipy.org/doc/scipy/index.html), [Transformer Reinforcement Learning](https://huggingface.co/docs/trl/en/index)

## 5. Technical Description:

In order to resolve the Brainteasers task, we have decided to use two large language models Phi 3.5 Mini and Mixtral 8x7b, since they have been trained on vast datasets that encompass various problem-solving techniques, heuristics, and logical reasoning strategies. As a result, these models can approach brainteasers from multiple angles, offering insightful results. Additionally, we have chosen 2 BERT models known as RoBERTa and DeBERTa as BERT models process text bidirectionally, meaning it considers both previous and subsequent words to determine meaning. This ability allows them to understand puzzles where context plays a critical role in arriving at the solution.

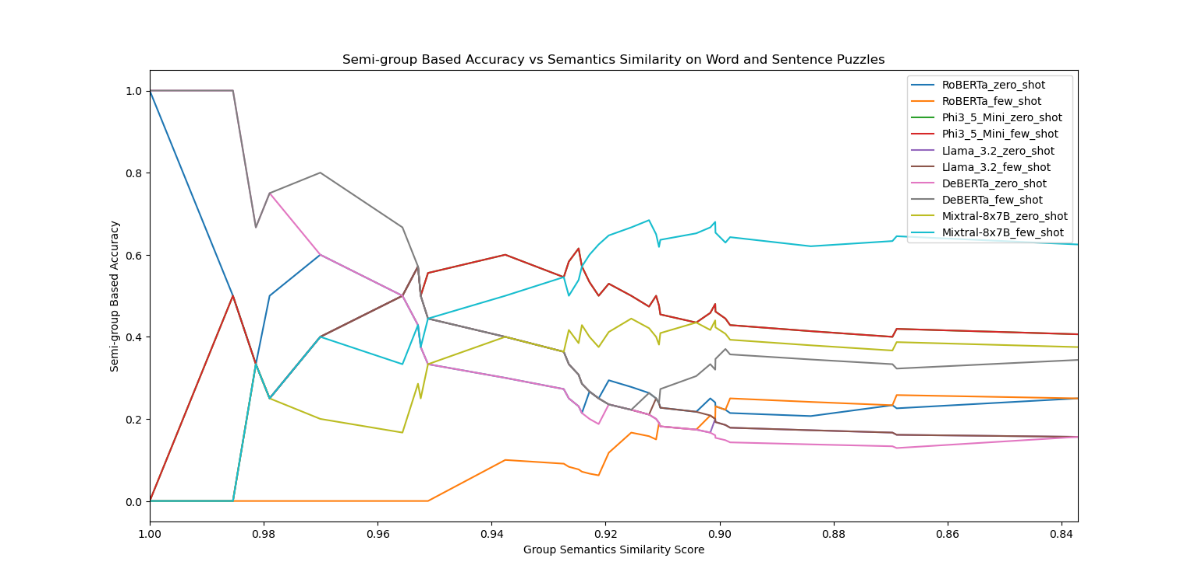
#### a. Prompt Engineering

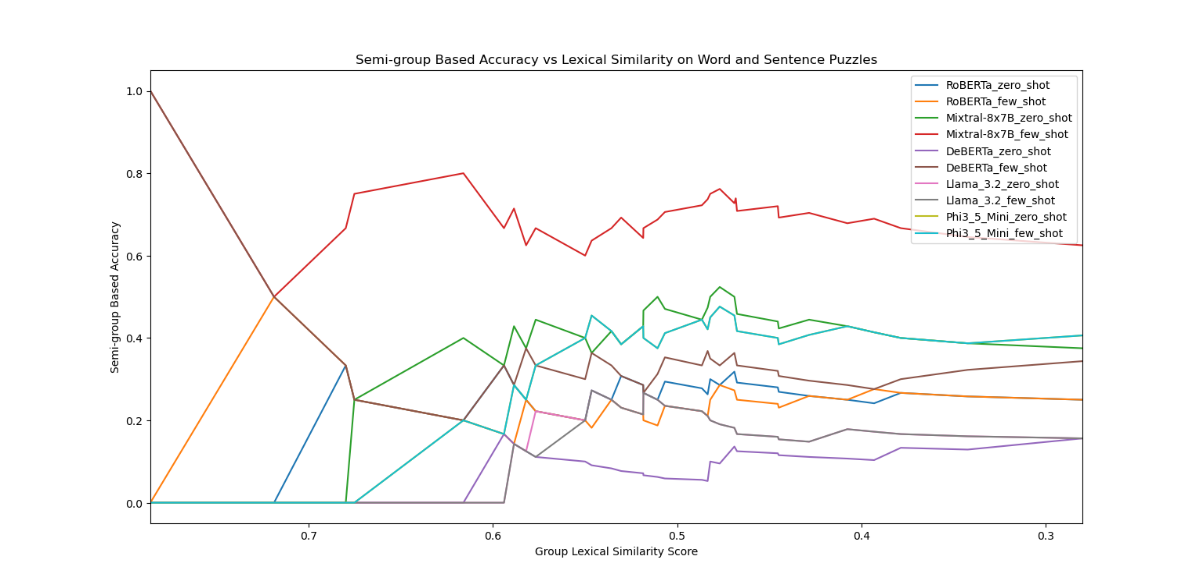
Regarding Large Language Models, for zero-shot prompting, we first create a system prompt to constrain each model’s answers which is “You are an assistant answering brainteasers and riddles for a test.” Then, we add the instruction prompt “Choose one of the following answers”. Finally, for each query, we add a question prompt which is a question from test set in a formatted as the question followed by the list of choices separated by a new line. For few-shot prompting, we add three random questions from the train dataset with each question’s choice list followed by “The correct answer is” and the correct answer. This information in added between after the instruction prompt but before the actual question prompt. For example, given the question “Imagine you are in a room, with no doors, windows, or anything. How do you get out?”, the prompts are formatted as:

|  |  |
| --- | --- |
| **Zero-shot prompting** | **Few-shot prompting** |
| You are an assistant answering brainteasers and riddles for a test  Choose one of the following answers.  Imagine you are in a room, with no doors, windows, or anything. How do you get out?  Stop imagining.  Jump out of the roof.  Break the Wall.  None of above. | You are an assistant answering brainteasers and riddles for a test  Choose one of the following answers.  What is the most fast city?  Urban city.  Inner city.  Velocity.  None of above.  The correct answer is Velocity.  What kind of a cup doesn’t hold water?  A teacup.  A cupcake.  A coffee cup.  None of above.  The correct answer is A cupcake.  Mr. and Mrs. Mustard have six daughters and each daughter has one brother. But there are only 9 people in the family, how is that possible?  Some daughters get married and have their own family.  Each daughter shares the same brother.  Some brothers were not loved by family and moved away.  None of above.  The correct answer is Each daughter shares the same brother.  Imagine you are in a room, with no doors, windows, or anything. How do you get out?  Stop imagining.  Jump out of the roof.  Break the Wall.  None of above. |

For BERT models, since they do not differentiate between different types of prompts, we are combining the system prompt, instruction prompt, and question prompt into a single prompt. Then, we evaluate the relevance score between each question’s choice and the query and choose the option with the highest relevance score.

One of our experiments is measuring the similarity between each base question and its semantic reconstruction form to determine the efficacy of the similarity score as an evaluation metric. We hope to implement this score as a stress testing metric to see if the model considers a question and its semantic reconstruction form as the same question. We calculated a semantic similarity score using MPNet model and a lexical similarity score using a Bag of Words model, using questions from the test dataset. Then we sorted the question groups in descending order of similarity score and plotted how the semi-group-based accuracy which only considers questions and their semantic reconstruction forms, of each model changes as the similarity score decreased. This metric’s efficacy is measured using Pearson Coefficients for each model.





|  |  |  |
| --- | --- | --- |
| Model | Semantic Similarity Coefficient | Lexical Similarity Coefficient |
| DeBERTa\_zero\_shot | 0.864559 | -0.49396 |
| DeBERTa\_few\_shot | 0.773355 | -0.14108 |
| Mixtral-8x7B\_zero\_shot | -0.71374 | -0.79943 |
| Mixtral-8x7B\_few\_shot | -0.83946 | -0.79943 |
| Phi3\_5\_Mini\_zero\_shot | -0.24504 | -0.56036 |
| Phi3\_5\_Mini\_few\_shot | -0.24504 | -0.58749 |
| RoBERTa\_zero\_shot | 0.744389 | -0.67722 |
| RoBERTa\_few\_shot | -0.90573 | 0.472837 |
| Llama\_3.2\_zero\_shot | 0.289425 | -0.6791 |
| Llama\_3.2\_few\_shot | 0.271278 | 0.238513 |
| Total Average | -0.0006 | -0.40267 |

From these results, for semantic similarity, we can see that there is strong positive correlation between similarity between a question and its semantic reconstruction form for DeBERTa and RoBERTa models. In contrast, other models show little to no correlation for this metric. On the other hand, several models show a positive correlation between Lexical Similarity and group-based accuracy. Thus, we can conclude that semantic similarity is a possible metric for BERT models while lexical similarity is a possible useful metric when training models. However, since we are receiving mixed results from this test, we have concluded that this metric should not be used as a final evaluation metric.

#### b. Hyperparameter tuning

It was required for us to fine tune the LLM model to suit the Brain-Teaser dataset. To this end, at first, we trained the model on the training dataset and measured the train loss, validation loss across each step in all the epochs. In the hyper parameter tuning process, we also had to take care of possible overfitting and underfitting of the model. They were measured by the divergence score which is the subtraction result of training loss from validation loss at the end of each epoch. If the divergence score was greater than 0, model was overfitting and if it was less than 0, model was underfitting. A divergence score of exactly 0 would indicate finding a good fit model. The greater or less the divergence score, the more was the degree of overfitting or underfitting.

For fine-tuning, we used Low-Rank Adaptation (LoRA), which introduces smaller trainable matrices alongside the model’s weight matrices. This reduces the number of trainable parameters, making the process more computationally efficient. We used the following parameters for fine-tuning:  
Batch size: 16, Epochs: 5, Rank: 4, LoRA rank: 4, LoRA Alpha: 8, LoRA Dropout: 0.1

We performed a grid search on the following values for learning rate and weight decay:

* Learning rate: [0.1, 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00001]
* Weight decay: [0.1, 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00001]

## Mixtral-8x7B-v0.1 Instruct:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  | Mixtral-8x7B obtained very high instance-based accuracy on the test dataset for almost all variants, especially, at smaller learning rates & weight decays. The loss divergence heatmap indicates that the model tends to overfit around learning rates of 1e-4 to 1e-3. |

## Phi 3.5 Mini:

|  |  |  |
| --- | --- | --- |
|  | A screenshot of a game  Description automatically generated |  |
| A screenshot of a computer  Description automatically generated | A screenshot of a computer  Description automatically generated | The Phi 3.5 model didn’t differ in performance across various learning rates and weight decays and had a consistent 63-65% accuracy for all instance-based tests. All variants of the model displayed an accuracy of 43.75% for group-based accuracy. The loss divergence heatmap also indicates that the fine-tuning had encountered an exploding gradient, resultant in non-existing data entries for certain instances. |

## Roberta:

|  |  |  |
| --- | --- | --- |
| A screenshot of a graph  Description automatically generated | A screenshot of a computer  Description automatically generated |  |
|  |  | At learning rates of 1e-3 and 1e-4 many instances of Roberta had performed the highest for instance-based accuracy. All the model-variants obtained very low group-based accuracy, failing to perform on context and semantic reconstructions effectively. The loss divergence graph indicates most models either fit well or slightly underfit on the train dataset. |

## Deberta V2:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  | The DeBerta model has performed the worst among all fine-tuned models with highest accuracy of 22% in both instance and group based with zero and few shots. For the DeBerta, all the loss values indicate underfitting apart from one where the learning rate was 0.001 and weight decay 0.0001. |

## Llama 3.2:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  | The Llama 3.2 model with 3B parameters was able to capture the semantic and contextual information of the datasets quite accurately. Hence the accuracy scores across all the instance and group based for zero and few shot prompts are on the high scale. Also, the model was closer to good fit in most of the learning rate and weight decay combinations. |

#### c. Challenges

In terms of challenges, we had difficulty with the different methods of tokenization each model had, how each model had different instruction formats. We also had to format the data according to the model in the correct choice order in the dataset. In addition, the outputs from the models had various differences in formatting utilizing commas and apostrophes in random places that makes it harder to get the correct accuracy metrics and requires further data post-processing. While it was important to perform some post-processing on the generated answers, it wasn’t easy to come up with a pattern for post-processing because each models’ generated answers contained different tokens, extra spaces, symbols and sometimes distractors as well. Also, we did face challenges in fine-tuning models, especially with the Llama model because fine-tuning Llama required supervised Fine-tuning Trainer (SFT Trainer).

## 6. Evaluation:

#### Selecting Best Model:

|  |  |  |
| --- | --- | --- |
| A screenshot of a computer screen  Description automatically generated | A screenshot of a computer  Description automatically generated | A screenshot of a computer  Description automatically generated |
| A screenshot of a computer  Description automatically generated |  | After finetuning all the models, we selected the best model by calculating total average accuracy for zero-shot and few-shot in both instance-based and group-based metrics, then normalized them using min-max normalization. The absolute value of loss divergence is also normalized. The best model score is then calculated by taking the difference between the normalized total accuracy and normalized absolute loss divergence. This score will take into account both the accuracy and over fitness of the model. |

**Base Model & Best Fine-tuned Model results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Zero-shot Instance | Few-shot Instance | Zero-shot Group | Few-shot Group |
| Deberta\_lr(1e-3)\_wd(1e-4) | 40.63% | 45.83% | 12.50% | 18.75% |
| Roberta\_lr(1e-4)\_wd(1e-3) | 25.00% | 42.71% | 0.00% | 15.63% |
| Mixtral\_lr(1e-5)\_wd(5e-4) | 90.63% | 100.00% | 71.88% | 100.00% |
| Phi\_lr(1e-5)\_wd(1e-5) | 64.58% | 64.58% | 43.75% | 43.75% |
| Llama\_lr(0.0001)\_wd(0.0001) | 87.50% | 87.50% | 81.25% | 81.25% |
| Deberta\_base | 23.96% | 41.67% | 6.25% | 18.75% |
| Roberta\_base | 31.25% | 29.17% | 9.38% | 9.38% |
| Mixtral\_base | 45.83% | 65.62% | 28.12% | 56.25% |
| Phi\_base | 60.42% | 60.42% | 34.38% | 34.38% |
| Llama\_base | 28.12% | 26.04% | 15.60% | 12.50% |

In order to measure the performance of each model, we used two metrics: Instance-based accuracy and Group-based Accuracy. Instance-based accuracy takes into account each  
individual question and calculates the accuracy based on the total number of correct answers. On the other hand, Group-based Accuracy considers each question and its associated adversarial instances as a group and calculate the accuracy based on the number of correct questions group, in which all the questions in a group must be correct for a question group to be considered correct.

From the charts, we can observer that Mixtral 8x7B at learning rate of 1e-5 and weight decay of 5e-4 model performed the best on our test dataset with 100% accuracy on both instance-based and group-based accuracy in few-shot. The BERT models significantly underperformed comopared to the larger-more complex LLM models, even after fine-tuning, and especially on group-based accuracy tests.

7. Error Analysis:  
One key point to address in the error analysis is the low scores for the BERT model compared to Mixtral, Phi 3-5 and Llama models. One key reason behind this can be explained with the loss divergence. All the values indicate underfitting and since Bert models have much less parameters to train and fine tune on, they can’t capture the semantics and contextual representations of the questions properly. This results in the BERT models receiving especially low group-based accuracies. With the Llama model, it was observed that the performance on the word problems was better compared to the sentence problems especially when sentences contained too many details and were lengthier.

## 8. Contributions:

* **Autri**: Implemented the base and fine-tuned Mixtral-8x7B models. Conducted evaluation of models based on instance-based accuracy and group-based accuracy. Performed preliminary testing while processing model data. Conducted final model accuracy comparisons for base and best-variants of each fine-tuned model.
* **Hung**: Implemented the framework to use Phi 3.5 Mini, RoBERTa, and DeBERTa models and measure semantic similarity using MPNet model. Build chart to measure the relationship between group-based accuracy and similarity scores. Cleaned and preprocessed files for data analysis.
* **Jawad**: Implemented the base and fine-tuned Llama 3.2 models. Calculated the cosine similarity among the questions and plotted the group-based accuracy as a function of increasing cosine similarity (Lexical Similarity) among the questions. From the intermediate report, got rid of the cosine similarity metrics to post-process the generated answers. Instead focused more on structuring the prompts with the correct formatting, instructions and parsed the excess tokens and lengthy sequences generated by the Llama models answers to preserve the actual answer. Analyzed the performance of the Llama model for lengthy sentence problems and fixed the lora and training parameters accordingly and cleaned the generated answers.