Preliminary Results

Project: Ensembling with BERT and its variations for Question Answering

Team:
Group 5 - Akash Govindarajula and Jakub Wasylkowski
Group 14 - Unnati Parekh and Nimesh Tripathi

Dataset and Statistics

The Stanford Question Answering Dataset (SQuAD)[1] 2.0 is a reading comprehension dataset, with reading passages from Wikipedia articles covering a diverse range of topics across a variety of domains, from music celebrities to abstract concepts. A passage is a paragraph from an article, and is variable in length. Each passage in SQuAD has accompanying reading comprehension questions. These questions are based on the content of the passage and can be answered by reading through the passage. The answer to every question is a segment of text, or *span*, from the corresponding reading passage, or the question might be unanswerable.

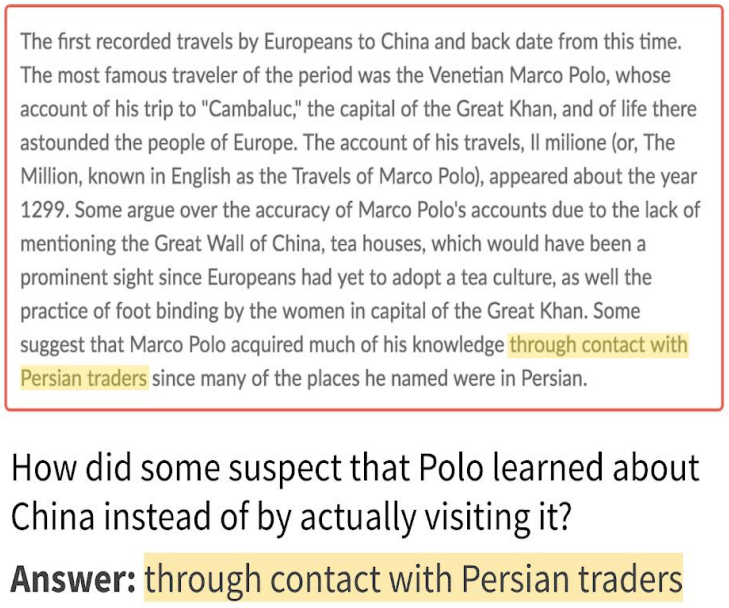

Figure 1: Example of SQuAD QA with highlighted answer text span in the passage

This is quite a challenging setup, with the possibility of a diverse range of questions that can be asked in the span setting. Rather than having a list of answer choices for each question, systems must select the answer from all possible spans in the passage, thus needing to cope with a fairly large number of candidates.

[1] https://rajpurkar.github.io/SQuAD-explorer/

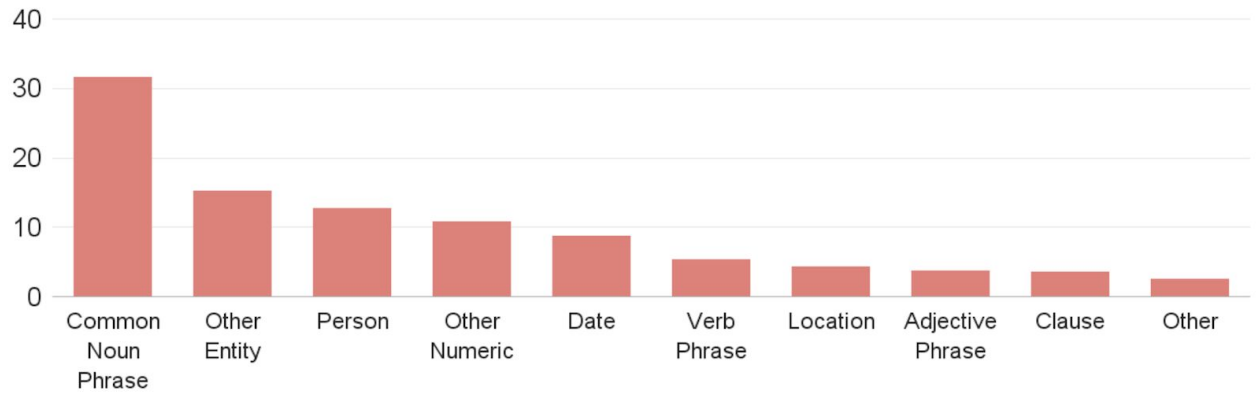

Figure 2: Distribution of answer types

The dataset consists of publicly available train and dev sets. The test set is hidden, results on which can be obtained when the model is uploaded to the official leaderboard to be evaluated against the official evaluation script. Below are the dataset statistics:

Dataset	**Total examples**
Train	130,319
Dev	11,873
Test	8,862

Table 1: SQuAD dataset statistics

The dataset is in the form of json object entries with the following structure:

- Paragraph title
- Context paragraph/text
- List of questions with associated list of answers

Each answer includes the start token for the answer pointing to the context string.

Model implementation

We have designed a set of ensemble approaches for combining the variations of BERT question answering models from the huggingface library[2]. We have used the following three models[3] for our ensemble:

1. BERT Mini (L=4, H=256)
2. BERT Small (L=4, H=512)
3. BERT Medium (L=8, H=512)

L = transformer layers, H = hidden embedding sizes

These models have been fine tuned on SQuAD 2.0 dataset for question answering downstream

[2] https://huggingface.co/transformers/index.html
[3] Turc, Iulia, et al. "Well-read students learn better: The impact of student initialization on knowledge distillation."

task. The models come with their unique tokenizer mappings. The question and context are encoded using respective tokenizers and loaded into each model for predicting start/end tokens (i.e. span) to the final answer. These are represented through tensors of all tokens in the original context with associated likelihood scores.

We have explored using the model output scores (logits before Softmax) in two ways, using the scores as is (_scores suffix in Table 2) and using these scores as probabilities after applying the Softmax function to the output tensor (_prob suffix in Table 2). The start and end token output scores/ probabilities are combined using a product.

We aim to analyse the result of combining these models using different approaches and see how the ensemble can boost performance of individual models depending on how they perform on a particular question. We are working on combining the scores in the following ways:

1. Best model confidence (best_model_confidence): We select the predictions of the model which outputs the highest scores/ probabilities for the start and end tokens.

2. Weighted average of model outputs (weighted_average): We rank the models based on their individual F1 scores on dev set. Then we use these weights to select the best model output such that a higher score is given to the predictions of the model with a higher F1 score.

3. Guided random search on weights: This is a variation of the above method. In this we call a routine to learn the best possible weights for models which we use for the final prediction. We initialise weights for the models randomly every iteration and use this weight to predict answers for the dev set and calculate F1 score for the same. This procedure is repeated for a limited number of iterations and the weights that obtained the highest F1 scores over all iterations are used for final predictions on the test set.

Primary evaluation metrics

The evaluation metrics we are using are the **Exact Match (EM)** and **F1 score**.

EM is a binary measure of whether the system output exactly matches the ground truth answer.

F1 score is a harmonic mean of precision and recall:

$$F1\ score\ =\ 2 * (\frac{(precision * recall)}{(precision + recall)})$$

Where $precision\ =\ \frac{True\ positive}{(True\ positive + False\ positive)}$ and $recall\ =\ \frac{True\ positive}{(True\ positive + False\ negative)}$

Preliminary results and future work

Preliminary results are currently based on very limited data for script refinement purposes. We expect these results to improve as we begin testing on the whole dataset.

Method	EM	F1 score
best_model_confidence_scores	54.0	56.94
best_model_confidence_prob	54.0	58.36
weighted_average_scores	53.0	57.14
weighted_average_prob	55.0	59.14

Table 2: Exact match (%), F1 score for the different methods discussed. The results are for 100 examples from dev set.

Our preliminary results are not very bad. We can see that the probabilities give better results for the QA task. Also, as our ensemble technique becomes more specific in tuning the individual model performances on the specific task, the ensemble score improves. Along the same trend, we expect our third approach to perform even better than the Weighted average approach.

We also noticed the models struggling with the "no answer" scenario by proposing a possible answer more often than not. Although some of these answers are mentioned in the dataset under "plausible answers", the evaluation metric supplied by the SQuAD team appears to dismiss those entirely. For this, we aim to look into the performance of the models on answerable and unanswerable questions separately. This might give us an insight into how the ensembling approach can be tweaked to handle this case.

Benchmarks

The SQuAD website offers a leaderboard of the best performing models verified by the SQuAD team. But the main concentration of our analysis will be to compare against the results in the following table, which are the EM and F1 scores of the individual models in our ensemble on the full SQuAD test set.

Model	EM	F1 score
BERT_mini	56.31	59.65
BERT_small	60.49	64.21
BERT_medium	65.95	70.11

Table 3: Exact match (%), F1 score for the models considered for the ensemble. The results are for the complete test dataset.