THE PROBLEM

With more content than ever on the internet, it can be hard to tell what is true and what is not.

In order to aid with fact verification, I decide to build an automated fact checker using [2]. The model fine-tunes Google’s BERT for sequence classification in order to predict statements as true, false, or not enough info given a context passage.

The model achieved 61.2% test accuracy, outperforming comparable systems.

THE DATA

The data used was constructed from the Fever dataset [4].

Statements were matched with context passages from the English Wikipedia and each true/false/not enough info label.

The data was split into a training set of 10k examples and dev/test sets of 1000 examples each.

THE MODEL

The model consisted of finetuning the base Bert model for sequence classification [2].

Inputs were constructed by concatenating the statement and the context and truncating to a maximum sequence length of 64.

A single linear layer was added to the Bert model in order to predict one of three labels – true, false, not enough info – for the statement-context input pairs.

THE RESULTS

Given the limited resources, limited time, and immediate success of the model, I only ran training once with a batch size of 32, a learning rate of 2e-5, and a maximum sequence length of 64.

The model achieved 61.7% training accuracy and 69.2% test accuracy.

DISCUSSION

Two interesting results from the model are its handling of inputs labeled “not enough info” and its comparative accuracy.

The model met the former with extreme success, achieving 100% precision and recall on these examples. For the latter, its accuracy was compared to the best results from two fact checking tasks, Fever (88.8%) and the fake news challenge (88.4%), and it outperforms both systems [12].

In fairness to the other models, these comparisons aren’t perfect as the tasks aren’t exactly the same. This lack of standardization is why a state-of-the-art model had not been established.

FUTURE

Given more time – and potentially resources – I could train on more data. Not only was training extremely time intensive, but the data and model were extremely large which caused me to quickly run out of storage, thus limiting the amount of training that could be done. To cope with this, only 1% of the potentially available data was used. I think using more of this data could improve the model by decreasing the variance between the training and testing accuracies.