Targeted Aspect Based Sentiment Analysis

Goal

The goal of this project is to predict the aspect and the sentiment of a given text and a defined target in the text

Hypothesis

By converting the given text into sentence pair QA dataset will help in Natural Language Inference (NLI) and summarizing the classification results can help in understanding the aspect

Dataset

Sentihood - Dataset for targeted aspect-based sentiment analysis (TABSA), which aims to identify fine-grained polarity towards a specific aspect. The dataset consists of 5,215 sentences, 3,862 of which contain a single target, and the remainder multiple targets.

Methodologies

1. Approach 1: Tried to encode the text to embeddings using sentence transformer and then perform hierarchical clustering to find the clusters. But the approach didn't turn out to be fruitful as the Adjusted Mutual Information score for the ground truth and predicted clusters was very less.
2. Approach 2: Finetune BERT language model for sentence classification task. The text is passed as the input and output label is sentiment. This approach gave good accuracy on sentiment (**0.94**) but it wasn't capturing the aspect.
3. Approach 3: Finetune BERT language model for sentence-pair classification task. Here the aspect information is stored in the form of question and the answer contains the sentiment.
4. Approach 4 [**TODO**]: Multi-label classification for the text. Where both the sentiment as aspect are the expected outputs for the text input

Analysis - Approach 3

The training accuracy of the model on the sentiment classification is ~94%. The validation accuracy is close to 90%. Overall the model was able to capture the sentiment information very well. The model was not directly evaluated for aspect (loss is not minimized over aspect).

The test results followed the same pattern as training. The accuracy over sentiments is 90% and the accuracy over aspects is 84%

Method for calculating accuracy for aspect:

The auxiliary sentences were created for all the unique aspects and if the model predicts a positive/negative sentiment for the auxiliary text then the aspect for that sentence is picked from the auxiliary question and evaluated with the ground truth.

Two training approaches were tried for the experiment:

1. Created auxiliary QA paris for all the aspects(12) and trained the model
2. Created QA pairs for top 4 aspects and trained the model.

The model had higher accuracy for sentiment in 2 but it wasn't able to generalize over all the aspects. Whereas model 1 was a little on the lower side in sentiment accuracy but it generalized well over all the 12 aspects. But the model tends to have false positives for the aspects

Pytorch:

I really like pytorch for my development because of its ease of use. But some things which I really wish pytorch supports are visualization board and deployment. Even though there's a workaround currently there isn't any direct tool to get TFBoard like visualization, I wish this changes soon. Although it's easy to wrap inference code in flask I wish pytorch has TF Serving like features for deployment environments.