Text Mining and Natural Language Processing

2023-2024

CLUELESS

Contextual **L**earning for **U**nderstanding **E**motion with **L**imited **E**xposure to **S**emantic **S**ymbols

By

Arib Zayn Araf. ID:514550

Mahfooz Anas. ID: 514587

Introduction

The project compares the performance of two pre-trained language models on predicting emotion labels on a popular emotion dataset, along with a new approach to handling of the raw dataset, to force the models to generalize better using context.

Data

The project applies the methodology on the GoEmotion dataset, taken from hugging face. The simplified version has been used because of the limited resources to train, which consists of 58,009 reddit comments, each marked with a single label out of 27 different labels, that has been used in the whole dataset. The dataset has been split into 3 separate sets for training, testing and validation, with 75% data(43,410 reddit comments) on training set, 9.35% (5426) on validation set and rest 9.35% on testing set. The project instead uses 90% of the dataset (every set has been reduced to 90% of its original size) to reduce computational burden.

Methodology

The main focus of the project is to fine-tune 2 pre-trained language models for emotions on the GoEmotion dataset for classifying the emotion labels of each data in the testing set. **What's new?** Inspired by how BERT makes better use of context by masking words, the method implements a masking strategy during the pre-processing phase, where words that explicitly show emotions are masked 10% of the time to improve generalization using surrounding context, instead of associating emotion labels by using certain obvious emotion words(Clues). Hence the name CLUELESS. The rest of the project follows the standard fine-tuning procedure. The main goal is to compare whether pre-trained language models **EmotionBERT** or **RoBERTa** do better at predicting labels for emotion. Before fine-tuning the model, 10% of the dataset has been used, along with a grid search over 3 different learning rates and 3 epochs for each, to find the optimal learning rate for our optimizer. A

small dataset has been used for limited resources, as grid search requires running the training 3 times for 3 learning rates.

Pre-Processing : In this step, the dataset has been loaded and reduced to 90% of its size using the reduce dataset function. An NRC text file has been added, which contains all words that directly describe emotions, and it has been used to create an emotion word list. Later , by using lexical resources from WordNet, an expanded emotion word list has been created, where the synonyms and derivational form of the emotion words are also added.

Masking : In this step, words in the dataset that also appears in the expanded emotion word lists are replaced with a [MASK] 10% of the time, thus forcing the model to use the context to generalize, instead of associating certain words with each emotions directly, like a shortcut, which will cause wrong predictions for sarcastic sentences etc. . The masking has been done using mask_emotion_words function. Another possible use of this method would be to use a list of words for a certain race or ethnicity, to prevent the model associating certain races with bad or good, in case those racial words appear more often with good or bad. Thus, reducing racial and societal bias in the model.

Tokenization : The texts in the datasets were tokenized by RoBERTa tokenizer and EmotionBERT on their corresponding codes.

DataLoading : This step used DataCollatorWithPadding and DataLoader to create dynamically padded batches of tokenized data for processing by PyTorch.

Training the model : Here a RoBERTa-based classifier and a EmotionBert-based classifer is used to train on the dataset, using the optimal learning rate(4e-5) we found out using grid search in smaller dataset. The batch size for both training and val sets used were 16,and the training ran for 2epochs. 2 epochs have been used as during the grid search using smaller dataset, the best validation loss was already found on first epoch. But that could have been caused because of small validation set, so instead of just 1, 2 epochs were used here for better evalulation. The training step uses an AdamW optimizer along with CrossEntropyLoss loss function from Pytorch. Validation loss has been calculated and printed after each epoch to check for overfitting.

Note : if training loss constantly decrease and validation loss inscrease in each epochs, the model is overfitting to training data.

Evaluation : The fine tuned model is applied on both test set and validation set to check its accuracy , weighted and macro f1 score, and other metrics etc.

Result and Analysis

	EmotionBert	RoBERTa
Accuracy	0.52	0.57
Macro F1 Score	0.40	0.42
Weighted F1 Score	0.51	0.55

The RoBERTta model performs better , as it has higher macro F1 score and weighted f1 score. As for accuracy, RoBERTa still has higher score, but accuracy is not a reliable metrics for these , as it can be biased towards dominant classes of labels. The macro F1-score is more important in these types of models because it evaluates the model's performance equally across all classes, regardless of class size, and by that metric, the performance of RoBERTa was better, but the values are too close. In conclusion, both model had competitive performance on this Goemotion dataset, but roBERTa performed slightly better.

Conclusion

The project tries to compare the performance of two pre trained model, roBERTa and EmotionBERT, with an emotion word masking strategy to force the models to use context to classify their emotion label on the goEmotion dataset. Grid search and multiple trial has

been used to find optimal parameters for both model and were evaluated according to their macro f1 score.

Contribution

Arib Zayn Araf:

*Proposed emotion-word masking strategy.

*Evaluated RoBERTa performance.

Mahfooz Anas:

* Suggested grid search for hyperparameter optimization.

* Evaluated EmotionBERT performance.

The 2 .ipynb files of both contributors were later merged using nbmerge to create a single .ipynb file.

AI policies

AI has been used to get suggestions on what can be a better loss function, optimizer and suggestions of hyperparameters that are usually used on models like these. The model has been tested on 2 optimizers and 2 loss functions. ChatGPT suggested adamW and adammac with recommendation for AdamW, which was correct. As for loss function, ChatGpt recommended using BCEWithLogitsLoss, but compared to using CrossEntropyLoss from torch, the performance was worse. The initial learning rate that was used in the grid search, 4e-5, 3e-5, 5e-5, were also suggested by ChatGPT. The prompt for these questions were pretty straight forward. For example : "hey chatGPT, I want to use RoBERTa to fine tune on the go emotion simplified dataset, what optimizers and loss functions would be my best choice? and recommend me the best ones between your suggestions". etc.