CapstoneProposal.md

Machine Learning Engineer Nanodegree

Capstone Proposal

Shekhar Prasad Rajak Jan 18, 2021

Proposal

Kaggle Problem: Tweet sentiment extraction

"My ridiculous dog is amazing." [sentiment: positive]

With all of the tweets circulating every second it is hard to tell whether the sentiment behind a specific tweet will impact a company, or a person's, brand for being viral (positive), or devastate profit because it strikes a negative tone. Capturing sentiment in language is important in these times where decisions and reactions are created and updated in seconds. But, which words actually lead to the sentiment description? In this competition you will need to pick out the part of the tweet (word or phrase) that reflects the sentiment.

Help build your skills in this important area with this broad dataset of tweets. Work on your technique to grab a top spot in this competition. What words in tweets support a positive, negative, or neutral sentiment? How can you help make that determination using machine learning tools?

In this competition we've extracted support phrases from Figure Eight's Data for Everyone platform. The dataset is titled Sentiment Analysis: Emotion in Text tweets with existing sentiment labels, used here under creative commons attribution 4.0. international licence. Your objective in this competition is to construct a model that can do the same - look at the labeled sentiment for a given tweet and figure out what word or phrase best supports it.

Domain Background

There is classical sentimental anslysis problem, where we have to check if the context have positive, negative or neutral sentiment. With this problem, we are trying to find the consecutive words that makes the whole context positive/negative or neutral.

BERT and it's variants like RoBERTa and ALBERT are helping in most of the common NLP problems and one of them is Question/Answering.

BERT is language representation model, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models, BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

Using BERT we can try to extract the phrase from the context that tends to particular sentiment.

Problem Statement

In dataset, we have :

```
textID - unique ID for each piece of text
text - the text of the tweet
sentiment - the general sentiment of the tweet
```

and the model will find the selected_text - the text that supports the tweet's sentiment.

More details: Tweet sentiment extraction

Datasets and Inputs

From the kaggle data:

Files we have:

train.csv, test.csv, and sample_submission.csv.

What should we expect the data format to be?

Each row contains the text of a tweet and a sentiment label. In the training set you are provided with a word or phrase drawn from the tweet (selected_text) that encapsulates the provided sentiment.

Make sure, when parsing the CSV, to remove the beginning / ending quotes from the text field, to ensure that you don't include them in your training. What am I predicting?

You're attempting to predict the word or phrase from the tweet that exemplifies the provided sentiment. The word or phrase should include all characters within that span (i.e. including commas, spaces, etc.). The format is as follows:

```
<id>,"<word or phrase that supports the sentiment>"
```

For example:

```
2,"very good"
5,"I am neutral about this"
6,"bad"
8,"if you say so!"
etc.
```

Files

```
train.csv - the training set
test.csv - the test set
sample_submission.csv - a sample submission file in the correct format
```

Columns

```
textID - unique ID for each piece of text
text - the text of the tweet
sentiment - the general sentiment of the tweet
selected_text - [train only] the text that supports the tweet's sentiment
```

Solution Statement

Steps:

- Cleanup the training data
- Prepare the tokenizer (planning to use ALBERT tokenizer from transformer library of huggingface)
- Get the encoding and customize the ALBERT model of the transformer for this problem statement.
- Overiride the forward pass, evaludation steps
- Hypertuning the model parameter and optimizer
- Check in each epochs the loss function value
- Get the performance for the train and test data using word-level Jaccard score.

Benchmark Model

There is lots of solution for this problem in kaggle dashboard: https://www.kaggle.com/c/tweet-sentiment-extraction/leaderboard

I want to compare the existing BERT solutions with the above idea of ALBERT. The accuracy (in terms of similarity using jaccard score) must be better than 70%, as we can see in leader board some of them using BERT in different ways and coming close to 0.7 score. So this variant should word in similar way but using specifically ALBERT.

I want to mainly compare (or work on top of) these 2 concepts :

- Twitter sentiment Extaction-Analysis,EDA and Model
- roberta inference 5 folds

I am trying to get better and optimized solution using ALBERT model which is optimized version of the BERT. Using huggingface library ALBERT model, we can try out different way of checking similarity between tokens/words and map the sentiments to words that is present in context.

Evaluation Metrics

The metric in this competition is the word-level Jaccard score. A good description of Jaccard similarity for strings is here.

A Python implementation based on the links above, and matched with the output of the C# implementation on the back end, is provided below.

```
def jaccard(str1, str2):
    a = set(str1.lower().split())
    b = set(str2.lower().split())
    c = a.intersection(b)
    return float(len(c)) / (len(a) + len(b) - len(c))
```

The formula for the overall metric, then, is: score=1n∑i=1njaccard(gti,dti)

where:

- n=number of documents
- jaccard=the function provided above
- gti=the ith ground truth
- dti=the ith prediction

Project Design

Once we cleanup the Text data and tokenize it using the appropriate tokenizer, we will use it as the training dataset.

- For cleaning up and understanding the data we will be having mutliple methods for exploratory data analysis and graphs.
- We will check the starting and ending index for the selected text in the original tweet text and store it in training dataset.
- Tokenize the data, masking and transformer concepts to be applied in the origina tweet data in training dataset.

`class TweetDataset(torch.utils.data.Dataset)` will be having the tokenized data.

`class TweetAlbertModel(transformers.AlbertPreTrainedModel):)` will be customized evaluation, forward pass and init methods.

Here is the skeleton for training step:

```
train_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)

optim = AdamW(model.parameters(), lr=5e-5)

for epoch in range(3):
    for batch in train_loader:
        optim.zero_grad()
        input_ids = batch['input_ids'].to(device)
        attention_mask = batch['attention_mask'].to(device)
        start_positions = batch['start_positions'].to(device)
        end_positions = batch['end_positions'].to(device)
        outputs = model(input_ids, attention_mask=attention_mask, start_positions=start_positions,
end_positions=end_positions)
        loss = outputs[0]
        print(loss)
        loss.backward()
        optim.step()
```

Once loss fucntion is pretty stable and we are getting occuracy, we will compare with leaderboard solutions and notebooks in kaggle.

Before submitting your proposal, ask yourself. . .

- Does the proposal you have written follow a well-organized structure similar to that of the project template?
- Is each section (particularly **Solution Statement** and **Project Design**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?
- Would the intended audience of your project be able to understand your proposal?
- Have you properly proofread your proposal to assure there are minimal grammatical and spelling mistakes?
- Are all the resources used for this project correctly cited and referenced?