Natural Language Processing

Supervisor: Dr. Seyed Saleh Etemadi

News Classification

Author:

Elnaz Rezaee elnazrezaee80@gmail.com

2023/2024

Contents

1 Introduction

In this section, we want to extract features from the data we extracted in the previous step.

2 Directory structure

```
- ReadMe.md
- Phase1-Report.pdf
- src/
  - (Project code files)
- data/
  - raw/
    - (Raw data files separated by labels, e.g., source1_true.csv)
  - clean/
    - (Cleaned data files, e.g., clean_data.csv)
  - wordbroken/
    - (Data separated by words, e.g., wordbroken_data.csv)
  - sentencebroken/
    - (Data separated by sentences, e.g., sentencebroken_data.csv)
- stats/
  - (Calculated information for the project data)
- experiments/
  - <experiment_name>/
    - (Working directory for the experiment)
- run_phase2.py/bat/sh
- run.log
- logs/
  - (Logs for each command/task/work)
- models/
  - (Trained models with unique names)
- latex/
  - (Text of the report in Farsi or English)
- Phase2_Report.pdf
```

3 Word2Vec

In this report, we describe the process of training word vectors using Word2Vec for each data category. We save the output models in the `word2vec` folder of the `models` directory.

The word vector models are saved in the `word2vec` folder of the `models` directory with the following filenames:

- `true.word2vec.npy`
- `mostly-true.word2vec.npy`

- `half-true.word2vec.npy`
- `barely-true.word2vec.npy`
- `false.word2vec.npy`
- `pants-fire.word2vec.npy`

3.1 Compare and analyze vectors of common words

We also compare the word vectors of common words across all categories using cosine similarity and categorize them as having the same vectors or different vectors based on a similarity threshold of 0.8.

After comparing the word vectors of common words across all categories, we obtained the following results:

- Common words with the same vectors:
  - 'increase'
  - 'men'
  - 'Democrats'
  - ...
- Common words with different vectors:
  - 'half'
  - 'Anthony'
  - 'annual'
  - ...

The similarity or difference in word vectors can be attributed to several factors. Here are some possible causes:

1. Contextual Differences: Different meanings or associations based on the specific context in each category.
2. Category-Specific Terminology: Unique terminology or jargon in each category.
3. Bias in Training Data: Biased or imbalanced training data.
4. Noise in Data: Inaccurate or noisy data.
5. Training Parameters: Different parameter settings during training, such as vector size and window size.

3.2 Train Word2Vec model on all data

In this part, we describe the process of training a Word2Vec model on all the data categories combined. The resulting model is saved as ”all.word2vec.npy” in the designated directory.

The training process was successful, and the Word2Vec model for all labels was saved at the designated location: ”models/word2vec/all.word2vec.npy”.

4 Tokenization

In this section, we describe the process of tokenization using the SentencePiece library with different tokenizer sizes. We evaluate the tokenization results on cleaned raw data and select the best tokenizer size based on the percentage of unknown tokens (Unk tokens). The best tokenizer model is saved, and the evaluation results are stored for further analysis.

The table below summarizes the evaluation results for each tokenizer size:

Size	Unk Percentages
100	2.1
500	3.8
1000	5.2
5000	7.5

Based on the evaluation results, the tokenizer with a size of 100 demonstrated the lowest average Unk percentage, making it the best tokenizer size for the dataset.

5 Language model

In this section, we describe the process of fine-tuning language models for each data category. We select a language autoregression model, fine-tune it separately for each category, and save the models in the language folder. We then generate a number of sentences for each category and save them in the stats folder.

You can see the generated sentences for true category below:

”An exceptional 11-year-old boy displayed remarkable heroism by rescuing and saving the lives of two individuals on a single day.” ”The second presidential impeachment of Donald Trump stands out as the most bipartisan impeachment in American history, as lawmakers from both sides of the aisle came together to hold the president accountable for his actions.” ”Over the past decade, the rate of adults in the United States receiving a flu shot has remained below 50”During my college days, we would occasionally utilize a popcorn popper to fry squirrel as a unique culinary experience.” ”In 2005, Hillary Clinton cosponsored legislation aimed at imposing penalties, including imprisonment, on individuals who engaged in flag burning.”

6 Feature engineering

In this section, we explore two simple architectures for data classification and train them using different features. The first architecture receives and classifies all sentence features at once, while the second architecture processes features one by one. We train and evaluate each architecture separately for the following features: sentence length, word length, words, word bigram, word2vec, word2vec bigram, and ParsBERT/BERT vectors. The classification accuracy for the validation, test, and train data is recorded and presented in a graph and a table.

7 Model architecture

In this section, we combine multiple features from the previous section and explore different architectures for data classification. We train, test, and compare at least three different architectures, including at least one based on Transformer, to determine the most effective architecture for the task.

We select the following features for the classification task: sentence length, word2vec, and word2vec bigram. These features have shown promising results individually and are combined in a desired way to enhance the classification performance.

8 Data augmentation

In this section, we utilized the GPT-Chat API from AI Open to generate data for different categories. We registered on the AI Open site, accessed the GPT-Chat API, and provided prompts for each category.

Due to the current API quota limitations, we encountered a RateLimitError while attempting to generate data using the GPT-Chat API. As a result, we were unable to complete the data generation process and conduct a thorough analysis of the generated data.

9 Rating by OpenAI

In this part like the previous section, we faced error due to limitation. So, I could not see the results, but I almost sure the code is correct.

Click here to see my github.

Click here to see my dataset.