Deep Learning Final Project

Andrew Chen

May 19, 2019

I pledge my honor that I have abided by the Stevens Honor System

1 Summary

I participated in a Kaggle competition called Quora Insincere Questions Classification and I worked by myself. The final methodology that I am using relies on ensembling 3 of the 4 provided word embeddings, then, I use a Bidirectional GRU layer, which compared to LSTM, according to Chung et al. 2014 provides similar performance but with less training time. After this, I performed normalization and applied a Dense layer to make a binary classification decision to determine if the input text was sincere or not

2 Problem Description

2.1 Problem:

The problem is to classify questions as being sincere or insincere based on the following criteria:

- Has a non-neutral tone
  - Has an exaggerated tone to underscore a point about a group of people
  - Is rhetorical and meant to imply a statement about a group of people
- Is disparaging or inflammatory
  - Suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype
  - Makes disparaging attacks/insults against a specific person or group of people
  - Based on an outlandish premise about a group of people
  - Disparages against a characteristic that is not fixable and not measurable
- Isn't grounded in reality
  - Based on false information, or contains absurd assumptions
- Uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers

2.2 Data:

The data provided are 2 files: `train.csv`, and `test.csv`. `train.csv` contains `qid` (question id), `question_text` (the question to classify), `target` (the label which is 1 or 0). `test.csv` contains `qid`, and `question_text`.

The task is to create a `submission.csv` that takes the questions in `test.csv` and give each question a label - 1 or 0. 0 represents a sincere question, and 1 is for insincere questions.

3 Solution

I did everything on Kaggle Kernel.

3.1 Model

1. Input Layer
2. Embedding Layer
3. Bidirectional GRU Layer
4. Global Max Pool 1D Layer
5. Dense Output Layer

I chose to use a GRU Layer instead of the LSTM Layer that we learned in class because according to the paper linked in the summary, GRU was faster to train, while still having good performance. I have looked at other sources which say that GRU's may be slightly worse at retaining long-term memory, but it seemed okay in this situation

With this configuration, I only needed two epochs before my model was unable to get any better.

The bigger choice in this problem was seeing what embedding I should use to maximize performance. The kernel provides you with 4 word embedding: Paragram, Glove, Wikinews, and GoogleNews. What I ended up doing was that I first made a baseline model to compare performance without word embedding, and then made another 3 for seeing performance of using 1 of the embeddings, and then lastly another one to compare performance of when we embed all of the results together. I did not implement the GoogleNews word embeddings because the kaggle kernel has a maximum RAM limit of 13GB. When I was running my kernel cells (with 3 of the 4 embeddings), I hit a maximum of around 12 GB of RAM. If I had implemented one more, I would go over the RAM limit of the kernel. However, even with only 3 of the 4, I found that ensembling the results of the 3 models had an improvement over just using one word embedding, although not by much (~1-2% improvement).

3.2 Settings

Because this is a binary classification problem, my loss function was binary cross entropy. The optimizer was `adam`.

Other Parameters:

- Embedding size: 300 (how big is each word vector)
- Max Features: 50000 (how many unique words to use in embedding layer)
- Max Length: 100 (Number of words in each question to use before cutting it off)

3.3 Advanced Tricks

1. Ensemble: I used emsembling on the word embeddings layer and found that it had a (1-2% improvement) over using just the Glove word embeddings which was the best of the word embeddings on this particular data set.

4 Comparison Methods

I implemented the following models and tested them against each other:

1. Random guess model
2. Naive Sentiment analysis model
3. NN-based model without pretrained embedding layer
4. NN-based model with Paragram Word Embedding weights
5. NN-based model with Glove Word Embedding weights
6. NN-based model with Wiki-news Word Embedding weights
7. Ensembled results from Models 4-6

5 Outcome

I did not participate in an active competition. My final results:

Attempt	Public Score	Private Score
1	0.62441	0.62923
2	0.64745	0.65736

5.1 Attempt One

I just used the Paragram word embedding.

5.2 Attempt Two

I used the Ensemble methodology.

6 Results

The top scores are in the 0.701 range. If this were an active competition, I would have placed 1234th in the private leaderboard, and 1242th in the public leaderboards