Toxic Comment Classification challenge on Kaggle.

René Ángeles
Udacity Capstone - Machine Learning Engineering

November 3, 2019

1 Project's domain background

NLP is on demand as it has found applications in the most diverse areas raging from language translation, sentiment analysis, entity recognition up to language models and text generators. Part of the reason for this explosion of this area comes from the data abundance. Such text data not only mined from knowledgeable sources as books but is mined from discussion forums such as twitter and reddit. Needless to say, comments arising from the latter contain not only valuable information but noise. This motivates the creation of classification tools and in this project we propose to review current classification techniques and its deployment via SageMaker.

Concretely, we will focus on the Toxic Comment Classification Challenge launched on Kaggle by the Conversation AI team [**?**] founded Jigsaw and Google (both a part of Alphabet) to improve online conversation. The reason to pick this problem is that it is complex enough to incorporate current tools in NLP classification such the state-of-the-art word embeddings and well models for text classification such as recurrent and convolutional neural networks.

2 Problem statement

Our problem consists in building a multi-headed model that assesses various aspects of comments such as toxicity, treats, insults and identity-based hate. For each such toxicity one should provide the probabilities that a given only comment belongs to each of the toxic classes.

3 Datasets and inputs

Jigsaw provides a training dataset of 160K comments, with binary labels corresponding to the following classes: Toxic, Severe toxic, Obscene, Threat, Insult or identity hate. Additionally, a test dataset with over 153K unlabelled test comments is provided to make a submission to the competition. Both label and unlabelled distribution were taken Wikipedia's talk page edits. Additionally, we shall use a pretrained word embedding that is included via Torchtext and we will further trained this, and other model layers, as required.

4 Solution statement

There is several competitive models developed in this competition. Rather than focusing on creating a model to reach the accuracy of top competitors, which may involve ensemble techniques as well as vast computational resources, we shall focus on the task of creating developing models that can be train, deployed and updated on SageMaker, emulating a production setting. To this end, I propose to use three well tested deep learning strategies on natural Language processing:

- Long short-term memory
- Gated recurrent units
- Convolutional neural networks

Of course, on the road we shall tackle text preprocessing and a choice of pretrained word embedding. We will use Pytorch as model build library as it is easy to train locally and it facilitates deployment on SageMaker.

5 Benchmark model

The metric of this multi-label Kaggle classification challenge uses the average of the Area Under the Receiver Operating Characteristic Curve (ROC AUC) to classify solutions. Currently, the team leading this competition scored team 0.98856%.

As explained above, we aim to emulate a production environment not only for this particular problem but applied to other similar text classification settings. Hence it is reasonable to demand our model reach the modest 97% ROC AUC, but creating code that is easily trained, deployed and updated via Sagemaker. Furthermore, our models should be versatile enough that further improvements are easily achivable and generalisable to other classification settings, e g. modifying the network architecture and hyper-parameters must be easy.

6 A set of evaluation metrics

Our performance evaluation metric will be the class averaged, ROC AUC. Roughtly speaking, it measures how a model's probabilities separate the positive classes from the negative classes and, crucially, is a metric that asses a model regardless of the threshold chosen to map probabilities into categories. In addition, our aim is that the model is compact enough to perform on laptop cpu in around one hour (without hyperparameter tuning). Our non-quantitative metric is to create a metric that can be deployed with SageMaker, which can be easily improvable and updated by further work.

7 An outline of the project design

We shall use the contents of the Udacity nanodegree contents [**?**], the course on Sequence Models by Andrew NG [**?**], and the Ben Trevett tutorial series on sentiment analysis [**?**] to follow the following program:

- Data Exploration (Pandas)
- Text preprocessing (Spacy [**?**])
- Choosing an appropriate word-embedding (Torchtext)
- Try different well established models such as LSTM, GRU, Convolutional Networks (Pytorch)
- Deployment via Sagemaker.