Machine Learning Engineer Nanodegree

Capstone Proposal

Tatiana Gaponova

February 14th, 2018

Domain Background

I would like to dive into NLP field and investigate the problem of Toxic Comment Classification. This problem is inherently important for the modern world where it is possible to easily reach much more people than it was only 15-20 years ago. Unfortunately, various kinds of social media do not only allow to bring people together, but they also provide a stage for threats, harassment and cyberbullying. In order to make online environment healthier and safer it is important to timely detect potentially harmful content. Perspective API already offers a model which can distinguish between toxic and nontoxic text (*toxicity is defined as a cause of someone's leaving a discussion*) but it cannot discriminate between different types of toxicity (threat, insult, obscene etc.) and can be further improved in terms of accuracy.

Problem Statement

The objective is to create a model that will build on previous work and provide some extended functionality. The final model should be able to process a chunk of text and output a probability of this text to be toxic discriminating between different types of toxicity:

1) general toxicity
2) severe toxicity
3) obscene
4) threat
5) insult
6) identity hate

Datasets and Inputs

The main dataset (provided by Jigsaw on the Kaggle platform) consists of over 100k comments from Wikipedia's talk page edits. These comments were manually labeled by 10 different annotators for toxic content of 6 categories (listed in the problem statement section). Given the

subjective nature of the problem some labels may seem questionable but the general tendency is lucid. Picture 1 provides a snapshot of the dataset structure.

Picture 1. Snapshot of the dataset

comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
I shalt sever thy head at the neck. It will be dreadfully slow and painful, in accordance with Wikipedia policies.	0	0	0	1	0	0
Die Whore Die you whore.	1	1	1	1	1	0
I can accept the topic ban and understa	0	0	0	0	0	0

This dataset is the main source for model development. However, it can be augmented by various third-party resources like lists of toxic words and thesaurus synonyms.

Solution Statement

The problem falls into multilabel classification framework. There are 3 main steps that should be accomplished:

1. **Text preprocessing.** Stopwords removal, lowercase conversion, punctuation removal, detection of misspellings (for example, with word2vec or fastText), lemmatization.
2. **Text vectorization.** Tf-idf vectorization and some other approaches are to be explored like word2vec.
3. **Model training.** Various options are available here as a lot of ML algorithms can be generalized to accommodate multilabel classification problem. One approach is one-vs-all method which involves training one binary classifier (for example, logistic regression) for each label. Neural networks can also be used to solve the problem: different NN architectures (which have 6 output neurons that correspond to the 6 types of toxicity) can be explored.

Benchmark Model

The benchmark model can be represented by the lines of python-style pseudocode below. Here we have simple tf-idf vectorization with no preprocessing and train 6 different logistic regression (LR) classifiers for each label which give us probability estimates. The performance of this model is quite good with the score on the test set equal to 0.97 (the metric is described in the next chapter).

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression

train, test = load_data()
vec = TfidfVectorizer()
```

```
trainX = vec.fit_transform(train.text)
testX = vec.transform(test.text)

for label in labels:   # ['toxic', 'severe_toxic', 'obscene', 'threat', 'insult', 'identity_hate']
   y = train.label
   model = LogisticRegression()
   model.fit(trainX, y)
   test.label = model.predict_proba(testX)
```

Evaluation Metrics

The performance for each toxicity type can be measured by ROC AUC, this metric shows how close our probability estimates are to the actual labels. There are 6 types of toxicity, so there will be 6 AUC scores, that can be averaged to get one score to represent model's quality.

Project Design

The benchmark model already gives good results. What can be done to improve the performance? The most obvious answer is to perform preprocessing and evaluate how it is going to influence the model:

1. Clean the text from stopwords, system characters and unnecessary punctuation.
2. Lemmatize the text.
3. Augment the text with the closest synonyms or similarly spelled words, this step may be especially useful as lot of words are intentionally misspelled (like, for example, 'sukcs', 'fuk' etc).

The second part is related to vectorization. In the benchmark model this task is carried by simple tf-idf, but other approaches can be tried here. For example, we can use word2vec model (pretrained or trained on the provided dataset) to take all word2vec vectors that constitute a comment and sum them up to get one single vector to represent this comment.

The third part is related to the algorithmic side of the project. LR model that was used in the benchmark can be further improved by tuning model's hyperparameters (the benchmark uses default python hyperparameters). This can be done by performing randomized search over possible LR hyperparameters. Other ML algorithms, like neural networks (RNN-LSTM) can be explored as well.