Machine Learning Nanodegree
Capstone Proposal

Sam Mottahedi

01/08/2018

1 Domain Background

Natural language processing (NLP) is one of the most important technologies of the information age. Understanding complex language utterances is also a crucial part of artificial intelligence. Applications of NLP are everywhere because people communicate most everything in language: web search, advertisement, emails, customer service, language translation, radiology reports, etc. There are a large variety of underlying tasks and machine learning models behind NLP applications.

Deep Learning (4) is another branch of machine learning that allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. Deep learning techniques have dramatically improved the state-of-the-art in areas such as speech recognition (2), image recognition (3) and natural language processing (1).

Recently, deep learning approaches have obtained very high performance across many different NLP tasks. These can solve tasks with single end-to-end models and do not require traditional, task-specific feature engineering

In this project a sequence classification model based on deep Recurrent network model is used to classify wikipedia comment as harmful. Such models can help developers or moderators improve conversations online by predicting the effect of comments on quality of conversions online. It can also work as feedback tool that inform commenters how they words are perceived by others.

2 Problem Statement

Free expression and sharing information is the greatest impact of internet in modern society. Unfortunately, online abuse and harassment can lead limit self expression on the web. Many platforms struggle to effectively facilitate conversations, leading many communities to limit or completely shut down user comments.

In this Kaggle competition , participant are challenged to build a multi-headed model that's capable of detecting different types of of toxicity like threats, obscenity, insults, and identity-based hate.

3 Datasets and inputs

The Kaggle competition dataset provided is Wikipedia Human Annotations of Toxicity on Talk Pages and contains $160,000$ human labelled annotations based on asking 5000 crowd-workers to rate Wikipedia comments according to their toxicity (likely to make others leave the conversation). Each comment was rated by 10 crowd-workers. The Test dataset which is used to evaluating performance in competition consist of $226,998$ unlabeled comments. Each comment in the training set can be labeled with 6 labels which are not mutually not exclusive. These 6 labels wit their prevalence are:

- Toxic (0.4300)

- Severe Toxic (0.0455)
- Obscene (0.2410)
- Threat (0.0144)
- Insult (0.2248)
- Identity hate (0.0384)

The data will be divided in to 3:1:1 for training validation and testing. Since the labels are not balanced, during the training each batch of data will be sampled using a Stratified sampling method to expose the model to different class labels.

4 Solution statement

Recurrent neural network architectures combining with attention mechanism, or neural attention model, have shown promising performance recently for the tasks including speech recognition, image caption generation, visual question answering and machine translation. In the sequence labeling tasks, the model input is a sequence, and the output is the label of the input sequence. The major difficulty of sequence labeling is that when the input sequence is long, it can include many noisy or irrelevant part. If the information in the whole sequence is treated equally, the noisy or irrelevant part may degrade the classification performance. The attention mechanism is helpful for sequence classification task because it is capable of highlighting important part among the entire sequence for the classification task.

In this project, each comment is treated as set of features using pre-trained GloVe word embedding. A Bi-directional recurrent neural network with Gated Recurrent Units (GRU). Attention mechanism is added on top of the Bi-directional RNN which improve the performance of the model focusing on important parts of long sequence and reduce the effect of noise and unrelated information. The classification task is a multi-label classification where comments toxicity labels are not mutually exclusive.

5 Benchmark model

The baseline chosen here in order to get better understanding of the problem from a general machine-learning perspective. To this end, the baseline model is logistic regression model based on Term Frequency-inverse document frequency data.

6 Evaluation metrics

Submissions are evaluated on the mean column-wise log loss. In other words, the score is the average of the log loss of each predicted column.

Since class labels are not mutually exclusive, a multi-label classification loss function is required here. Multi-label classification (MLC) is a prediction problem in which several class labels are assigned to single instances simultaneously as follows:

$$loss(\hat{y}, y) = \frac{1}{|L|} \sum_{l=1}^{l=|L|} -(y_l - log(\hat{y_l}) + (1 - y_l) \cdot log(1 - \hat{y_l})) \tag{1}$$

7 Project design

7.1 Programming language and Libraries

- Python 3.5
- Pandas
- Nltk
- Tensorflow (1.4)

7.2 Pre-Processing

An important part any natural language processing is cleaning and processing text for the use in any machine-learning algorithm. Pre-Processing step taken here are:

- Tokenizing and normalizing text and numbers using nltk package and regular expression.
- converting tokenized words to ids.
- Building Vocabulary of words included in the input dataset. It's necessary to limit the size of vocabulary since hardware limits the word embedding matrix dimension ($vocab_size \times word_embedding_dimension$) possible.

7.3 Model Definition

The Tensorflow model will be place in a Python class with necessary interface for creating the compute graph. Also, class methods are decorated with modified version of the property decorator since it's not desirable to create a new graph component each time the method is class during training to inference. The code Snippet below shows the basic structure of the model class and decorator.

```
import functools

def lazy_property(function):
        attribute = '_cache_' + function.__name__

        @property
        @functools.wraps(function)
        def decorator(self):
                if not hasattr(self, attribute):
                setattr(self, attribute, function(self))
                return getattr(self, attribute)

        return decorator
```

```
class Model:

def __init__(self, data, target):
```

```
                self.data = data
                self.target = target
                self.prediction
                self.optimize
                self.error

        @lazy_property
        def prediction(self):
                NotImplementedError
        @lazy_property
        def optimize(self):
                NotImplementedError
        @lazy_property
        def error(self):
                NotImplementedError
```

7.4 Attention Mechanism

The attention mechanism extract the important word in the sequence and aggregate the represetation of those word to form a sentence vector:

$$u_{it} = tanh(W_w h_{it} + b_w) \tag{2}$$

$$\alpha = \frac{exp(u_{it}^T u_w)}{\sum exp(u_{it} u_w)} s_i = \sum \alpha_{it} h_{it} \tag{3}$$

where h_{it} is the concatenated output of bi-directional layer.

7.5 Optimizer

The optimization algorithm used here is stochastic gradient decent with exponentially decreasing learning rate. The learning-rate is hyper parameter that needs to be tuned during training.

References

[1] R. Colbert et al. Natural language processing (almost) from scratch. In *Journal of Machine Learning Research*, volume 12, pages 2493–2537, 2011.

[2] G. Hinton et al. Deep neural networks for acoustic modeling in speech recognition. In *IEEE Signal Processing Magazine*, volume 29, pages 82–97, 2012.

[3] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems*, volume 25, pages 1090–1098, 2012.

[4] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, pages 436–444, 2015.