Machine Learning Engineer Nanodegree

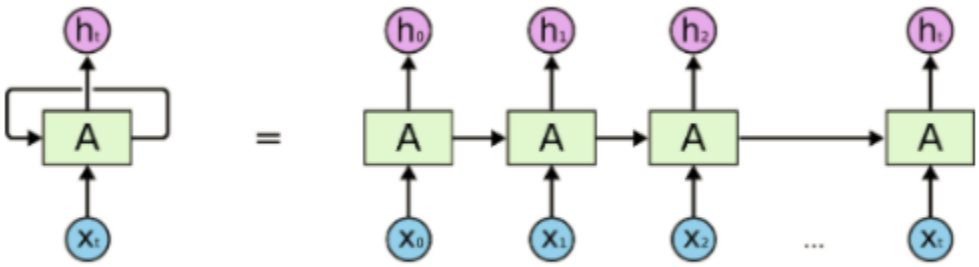
Capstone Proposal

Prateek Gupta
May 12th, 2018

Proposal

Domain Background

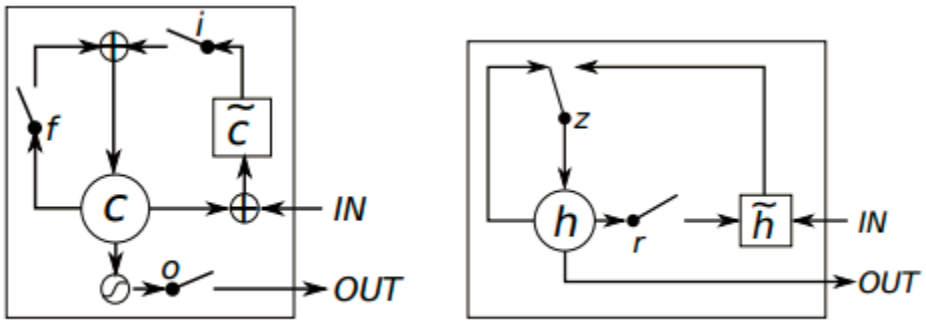
Natural-language processing (NLP) is an area of computer science and artificial intelligence concerned with analyzing, understanding, and deriving meaning from human language in a smart and useful way. Persistence of information is important for understanding context from a sentence. Traditional neural networks do not have this capability. RNN (Recurrent neural network) can address this issue by allowing information to pass through loops. In cases, where the gap between the relevant information and the place where it is needed is small, RNN can be used to learn past information.



An unrolled RNN

LSTM (Long Short Term Memory) is a special type of RNN, which can learn to connect the information with long term dependencies.

GRU (Gated Recurrent Unit) uses gating mechanism in RNN. The absence of output gate reduces the number of parameters as that of LSTM and provides better performance over LSTM over smaller datasets.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Problem Statement

The Conversation AI team is working towards improving online conversation.

Platforms struggle to effectively facilitate conversations among groups when there is a threat of abuse and harassment among people of different opinions.

The problem at hand is to identify and classify toxic comments during online conversation. This will allow platforms in finding which type of toxicity be allowed and which type of toxicity be discouraged.

Datasets and Inputs

The data is provided by Kaggle - https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data

File descriptions

1. Kaggle Files
    a. train.csv - the training set, contains comments with their binary labels
    b. test.csv - the test set
2. Additional Files
    a. FastText crawl 300d 2M – Collection of 2 million word vectors trained on Common Crawl
    b. GloVe Twitter - unsupervised learning algorithm for obtaining vector representations for words

Data Description

1. Columns in train.csv and test.csv
    a. Id – unique identifier

    b. Comment_text – comment by users
    c. Toxic – binary labels for toxic classification
    d. severe_toxic – binary labels for severe toxic classification
    e. obscene – binary labels for obscene classification
    f. threat – binary labels for threat classification
    g. insult – binary labels for insult classification
    h. identity_hate – binary labels for identity hate classification
2. In train.csv
    a. #Columns – 8
    b. #Rows – 159571
    c. This data is highly relevant as we need to train on this data to make classification
3. In test.csv
    a. #Columns – 2
    b. #Rows-153164
    c. This data is highly relevant as we need this data to test the performance of model on unseen data

Solution Statement

Manual tagging of toxicity cannot work at scale. And hence, we want to create a machine learning model that can be used to identify and classify toxic comments during online conversation. The deep learning model will be able to take new conversations as input, pass them through the network to provide a classification for the level of toxicity.

Benchmark Model

As this is a Kaggle competition, we will be submitting the solution via a late submission feature. Kaggle will bench mark our model against other models and provide a score. A personal goal would be to get a score 0.75+.

Evaluation Metrics

For evaluation, we will be using the roc_auc_score function under the sklearn library. Roc_auc_score computes the area under the Receiver Operating Characteristics Curve (ROC AUC) from prediction scores. The AUC values from 0.5-1 denotes an excellent classifier.

Project Design

We want to understand the different classifying of text in the input file. The difference between the different classifications can be used to create meaningful features.

We will use embedding layers instead of one hot encoding. One hot encoded vectors are high-dimensional as well as sparse. In a big dataset, this will not be computationally efficient. Embedding layers also provide the model to take relationships in language into consideration, which is not possible in one hot encoding.

As the data is generated by the user, data preprocessing will be important. Data preprocessing will include tasks like removing non-ascii characters, and fixing spelling mistakes using word2vec . After this, we can start creating features that will go into the model. To create embedding layers, we will be using a large corpus of data available from fasttext crawl and glove. If there is case of vanishing gradient, we will be changing the activation function. If there is a case of exploding gradient, we will be using gradient clipping. The models which we are planning to use are LSTM, GRU and bidirectional GRU.

References

1. Kaggle Toxic Comment Classification Challenge
   https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

2. A Beginner's Guide to Recurrent Networks and LSTMs
   https://deeplearning4j.org/lstm.html

3. Long short-term memory – Wikipedia
   https://en.wikipedia.org/wiki/Long_short-term_memory

4. Gated Recurrent Unit – Wikipedia
   https://en.wikipedia.org/wiki/Gated_recurrent_unit

5. Conversation AI
   https://conversationai.github.io/

6. FastText crawl 300d 2M
   https://www.kaggle.com/yekenot/fasttext-crawl-300d-2m

7. GloVe - https://nlp.stanford.edu/projects/glove/