Capstone Proposal

Eva Long

28 July 2018

Domain Background

Sentiment analysis of the Natural Language processing (NLP) field has been evolved quickly in the recent years. It helps us to understand the human attitudes and emotions towards a particular topic or product. Reliable results from such analyses can be of valuable information and good use in various domains such as social media and customer services.

The traditional method for sentiment analysis on texts may involve applying word embedding techniques such as bag of words, Vector Space model to extract data and data structure, and then adopting a supervised machine learning models such as Naive Bays and SVM. More recently, academic researchers have been exploring deep learning models for word embeddings and classifications, and it has proved to be very powerful. For instance, Researchers from Google developed word2vec which inputs corpus of documents and generates a vector space of hundred dimensions. (Mikolov, Tomas; et al., 2013) The new techniques are much faster to train and are able to identify words that have similar meanings and and are of the same context. Researchers also have shown that RNN (Recurrent Neural Network) and LSTM (Long Short Term Memory) performs better than other similar Neural network and classic supervised training models, as a recurrent structure learns from semantic of previous texts and can better capture contextual information. (Siwei Lai et al., 2015)

Problem Statement

I propose to predict the sentiments of texts on Twitter data (Tweets) by using a deep learning approach with Pre-trained Word2Vec embedding and a LSTM/RNN neural network classifier, to a traditional method with vector space model and a supervised machine learning model.

It will be a binary-class classification problem with 2 outcomes ‘ Negative’,’Positive’.

Datasets and Inputs

Dataset used in the analysis is the Sentiment 140 Twitter Data. It is obtained from Open source via the following website. http://help.sentiment140.com/for-students

The dataset contains 1.6 million twitter tweets collected in 2009. The data has six fields:

0 - the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)
1 - the id of the tweet (2087)
2 - the date of the tweet (Sat May 16 23:58:44 UTC 2009)
3 - the query (lyx). If there is no query, then this value is NO_QUERY.
4 - the user that tweeted (robotickilldozr)
5 - the text of the tweet (Lyx is cool)

The dataset is in cvs form and looks like this after converted to data frame using Pandas.

index	target	id	date	flag	user	text
1	0	146781067	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by texting it... and might cry as a result School today also. Blah!
2	0	146781091	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Managed to save 50% The rest go out of bounds
3	0	146781118	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire

Solution Statement

- Convert words to vectors using word2Vec using a pre-trained model, which creates an embedding layer.
- Feed into a LSTM and RNN network for the classification problem using Keras. I suspect LSTM would give better result as RNN can result in exploding or vanishing gradients during updates of the model weights as the information keeps feeding into the loop. LSTM addresses this issue by controlling memory process using a tanh and sigmoid layer. *
- Compile model to configure learning process

- Fit the model to training set and evaluate and results using validation and test set.

* Details in the Paper Long Short-Term Memory by Hochreiter, Schmidhuber (1997)

Benchmark Model

For the benchmark model, I will be using a bag of words model for feature extraction and a SVM/ Naive Bayes classifier with sklearn package. I propose to use the GridSearch function to find the best combination of parameters and classifiers. The result can be measured by various score metrics. (See evaluation metrics below)

Evaluation Metrics

The data will be split into training and test for the classification exercise and result will compare the F1 score of the on testing data for the proposed models and benchmark models. The reason for choosing F1 score over accuracy score is that the dataset can be imbalanced for the three labelled outcome.

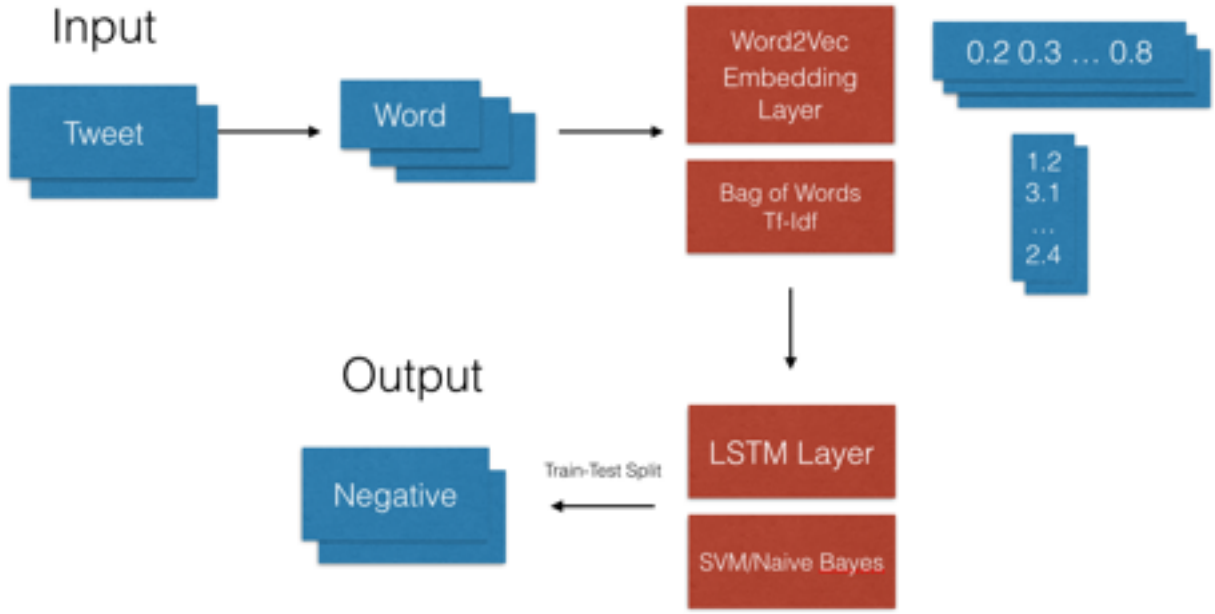
Precision = True Positive / (True Positive + False Positive)
Recall = True Positive / (True Positive + True Negative)

F1 Score= 2* (Precision * Recall)/(Precision +Recall)

Project Design

1. Data Preprocessing:
   Import data, clean columns and remove NAs, check data outcome distribution.
2. Feature Extraction:
   Vector-Space representation of the document using Tf-Idf and Pre-trained Word2Vec for benchmark and proposed solution. (using Genism/Sklearn library). Split data to train, validation and test set. (80%, 10%, 10%).
3. Deep learning layer:
   Feed data into LSTM/RNN and compiling the model using Keras.
4. Training and Validation:
   Fit train data to the deep learning layer; Fit the data to SVM/NB classifier and find best model using GridSearch in sklearn.
5. Cross Validation:
   Validate the model on the validation set for both models. Check the training and validation results for overfitting/under-fitting and tune the model parameters.

6. Prediction: Predict on test data and check the F1 score for both models.



References:

http://help.sentiment140.com/for-students
https://ahmedbesbes.com/overview-and-comparison-of-tranditional-and-deep-learning-models-in-text-classification.html
https://towardsdatascience.com/another-twitter-sentiment-analysis-with-python-part-9-neural-networks-with-tfidf-vectors-using-d0b4af6be6d7
https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5
https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9745/9552
http://www.bioinf.jku.at/publications/older/2604.pdf
http://www.volodenkov.com/post/keras-lstm-sentiment-p1/