ITCS 6156 MACHINE LEARNING

Mid Term Report

Quora Insincere Questions Classification
Detect toxic content to improve online conversations

Team Members

Member Name	Student ID
Akhil Morampudi	(801138186)
Bharadwaj Aryasomayajula	(801151165)
Mahanth Mukesh Dadisetty	(801034945)

Team Name: Invincible

Introduction

Summary

An existential problem for any major website today is how to handle toxic and divisive content. Quora wants to tackle this problem head-on to keep their platform a place where users can feel safe sharing their knowledge with the world.

Quora is a platform that empowers people to learn from each other. On Quora, people can ask questions and connect with others who contribute unique insights and quality answers. A key challenge is to weed out insincere questions -- those founded upon false premises, or that intend to make a statement rather than look for helpful answers.

Abstract

Machine learning models have been very successful when dealing with text classification problems. In this project we are tasked with classifying questions from a dataset provided by the popular website Quora, as 'sincere' or 'insincere'. The large-scale dataset, provided by the website Kaggle [https://www.kaggle.com/c/quora-insincere-questions-classification/data] contains over 1,300,000 questions with labels to train our models on. A separate test set that contains over 300,000 unlabeled questions is used by Kaggle to test our model. We implement three different models, logistic regression, Naive Bayes, and recurrent neural networks, and use different pre-trained word embedding's to achieve the best results

An insincere question is defined as a question intended to make a statement rather than look for helpful answers. Some characteristics that can signify that a question is insincere:[3]

- Has a non-neutral tone
- Has an exaggerated tone to underscore a point about a group of people
- Is rhetorical and meant to imply a statement about a group of people
- Is disparaging or inflammatory
- Suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype
- Makes disparaging attacks/insults against a specific person or group of people
- Based on an outlandish premise about a group of people
- Disparages against a characteristic that is not fixable and not measurable
- Isn't grounded in reality
- Based on false information, or contains absurd assumptions
- Uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers

The training data includes the question that was asked, and whether it was identified as insincere (target
= 1). The ground-truth labels contain some amount of noise: they are not guaranteed to be perfect

Literature Survey

A machine learning approach to sentiment analysis in multilingual Web texts [8]

Abstract:

Sentiment analysis, also called opinion mining, is a form of information extraction from text of growing research and commercial interest. In this paper we present our machine learning experiments with regard to sentiment analysis in blog, review and forum texts found on the World Wide Web and written in English, Dutch and French. We train from a set of example sentences or statements that are manually annotated as positive, negative or neutral with regard to a certain entity. We are interested in the feelings that people express with regard to certain consumption products. We learn and evaluate several classification models that can be configured in a cascaded pipeline. We have to deal with several problems, being the noisy character of the input texts, the attribution of the sentiment to a particular entity and the small size of the training set. We succeed to identify positive, negative and neutral feelings to the entity under consideration with ca. 83% accuracy for English texts based on unigram features augmented with linguistic features. The accuracy results of processing the Dutch and French texts are ca. 70 and 68% respectively due to the larger variety of the linguistic expressions that more often diverge from standard language, thus demanding more training patterns. In addition, our experiments give us insights into the portability of the learned models across domains and languages. A substantial part of the article investigates the role of active learning techniques for reducing the number of examples to be manually annotated.

Methodology:

Machine learning techniques for sentiment classification gain interest because of their capability to model many features and in doing so, capturing context, their more easy adaptability to changing input, and the possibility to measure the degree of uncertainty by which a classification is made. Supervised methods that train from examples manually classified by humans are the most popular. The most common approaches here use the single lowercased words (unigrams) as features when describing training and test examples.

In opinion mining certain sentiments are expressed in two or more words, and the accurate detection of negation is important because it reverses the polarity. Pedersen showed that word n-grams are effective features for word sense disambiguation, while Dave et al. indicated that they are able to capture negation. In an alternative approach to negation, each word following a negation until the first punctuation receives a tag indicating negation to the learning algorithm.

Other approaches select only a subset of the words, often by considering solely adjectives detected with a part-of-speech (POS) recognizer.

Pros:

1. The supervised techniques are applied mostly for recognizing the sentiment of complete documents.

2. Step-wise approach for the classification of texts by first removing objective sentences from it (using a machine learning-backed minimal cut algorithm), and then classifying the remaining ones.)

Cons:

1. Few weakly supervised learning approaches in the literature. In such settings the manual labeling is limited.Clustering can be used to determine the semantic orientation of many adjectives present in a corpus; the obtained clusters are then manually labeled.

2. Identify in texts product properties (PPs) and their correlated opinion words (OWs) by using an iterative cross-training method, which starts from a small labeled part of the corpus. Two naïve Bayes classifiers (respectively for detecting PPs and OWs) are trained using contextual features.

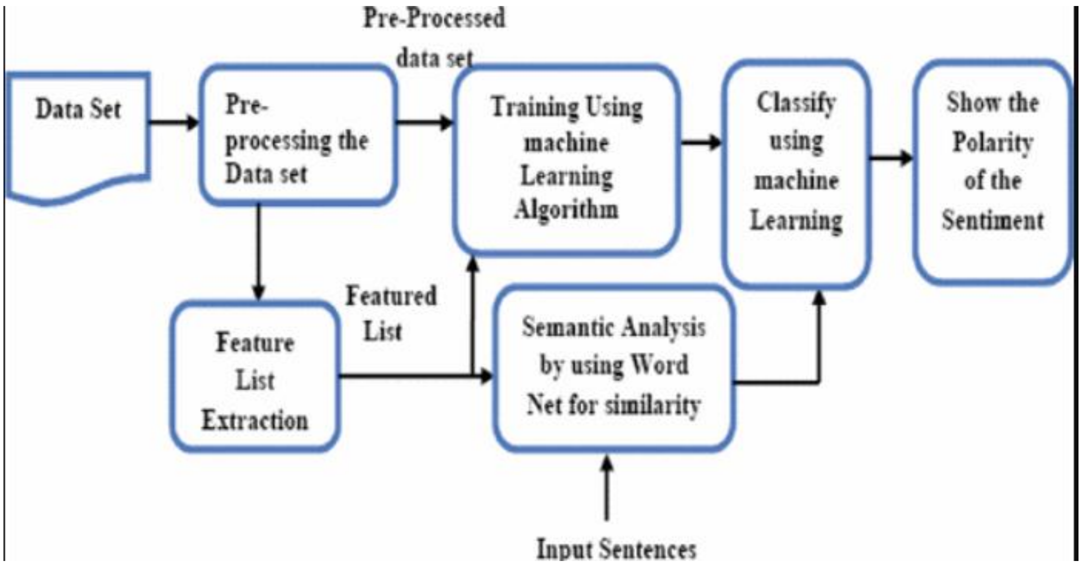
3. Noisy character of the input texts, the attribution of the sentiment to a particular entity and the small size of the training set.

Quora Insincere Question Classification:

Neural network models have been proved to achieve remarkable performance in text sentiment classification. CNN trained on top of pre-trained word vectors got great results for sentence-level classification tasks[8]. Some combinations and modifications can get improvement. Gated RNN dramatically outperforms than standard RNN[9] in document modeling for sentiment classification. A unified model with CNN and LSTM called C-LSTM is able to capture both local features of phrases as well as global and temporal sentence semantics.[10] [11].

Method

In our approach we used the Quora Insincere dataset and analyzed it. This analyses labeled datasets using the unigram feature extraction technique. We used the framework where the preprocessor is applied to the raw sentences which make it more appropriate to understand. Further, the different machine learning techniques trains the dataset with feature vectors and then the semantic analysis offers a large set of synonyms and similarity which provides the polarity of the content. The complete description of the approach has been described in next sub sections and the block diagram of the same is graphically represented in the below figure.


Pre-Processed data set
Data Set
Pre-processing the Data set
Training Using machine Learning Algorithm
Classify using machine Learning
Show the Polarity of the Sentiment
Featured List
Feature List Extraction
Semantic Analysis by using Word Net for similarity
Input Sentences

Objective

The objective of this project is to implement machine learning algorithms on the training dataset provided by Kaggle, and then create a set of predictions for a separate unlabeled test set. In this project we test several models that have been known to perform well with text classification problems. We begin with two baseline models, logistic regression and naive Bayes, and then create a more advanced recurrent neural network model.

Basic Feature Engineering:

We can add some features as a part of feature engineering pipeline for Quora Insincere Questions Classification Challenge.
Some features that I have included are listed below:

Data Pre-processing:

The text data is not entirely clean, thus we need to apply some data pre-processing techniques.

Pre-processing techniques for Data Cleaning:

- Removing Punctuation
- Number of capital letters
- Number of special characters
- Number of unique words
- Number of numeric
- Number of characters
- Number of stop words

Proposed Model and Methodology:

Machine Learning Methods:

Advance NLP Text Processing:[4]

Bag of Words (CountVectorizer)

Bag of Words (BoW) refers to the representation of text which describes the presence of words within the text data. The intuition behind this is that two similar text fields will contain similar kind of words, and will therefore have a similar bag of words. Further, that from the text alone we can learn something about the meaning of the document.

Word2Vec Feature (Word Embeddings)

Word Embedding is a language modeling technique used for mapping words to vectors of real numbers. It represents words or phrases in vector space with several dimensions. Word embeddings can be generated using various methods like neural networks, co-occurrence matrix, probabilistic models, etc.

Word2Vec consists of models for generating word embedding. These models are shallow two layer neural networks having one input layer, one hidden layer and one output layer.

Tokenizing:

Tokenization is the act of breaking up a sequence of strings into pieces such as words, keywords, phrases, symbols and other elements called tokens. Tokens can be individual words, phrases or even whole sentences. Tokenization is a necessary first step in many natural language processing tasks, such as word counting, parsing, spell checking, corpus generation, and statistical analysis of text. Tokenizer is a compact Python module for tokenizing, which converts Python text strings to streams of token objects, where each token object is a

separate word, punctuation sign, etc. It also segments the token stream into sentences, considering corner cases such as abbreviations and dates in the middle of sentences.
NLTK provides a number of tokenizers in the tokenize module

N gram:

In the fields of computational linguistics and probability, an n-gram is a contiguous sequence of n items from a given sample of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The n-grams typically are collected from a text or speech corpus. When the items are words, n-grams may also be called shingles.

For example, for the sentence "The cow jumps over the moon".
If N=2 (known as bigrams), then the n-grams would be:

- the cow
- cow jumps
- jumps over
- over the
- the moon

If N=3, the n-grams would be:

- the cow jumps
- cow jumps over
- jumps over the
- over the moon

In this we are using N-gram ranging from 1 to 5, so that we attains maximum efficiency while training the mode.

CountVectorizer:

CountVectorizer implements both tokenization and occurrence counting in a single class and converts a collection of text documents to a matrix of token counts.

Stop words are words like “and”, “the”, “him”, which are presumed to be uninformative in representing the content of a text, and which may be removed to avoid them being construed as signal for prediction. Sometimes, however, similar words are useful for prediction, such as in classifying writing style or personality.

If stop_words= ’english’, a built-in stop word list for English is used.
If a list, that list is assumed to contain stop words, all of which will be removed from the resulting tokens. Only applies if analyzer == 'word'.

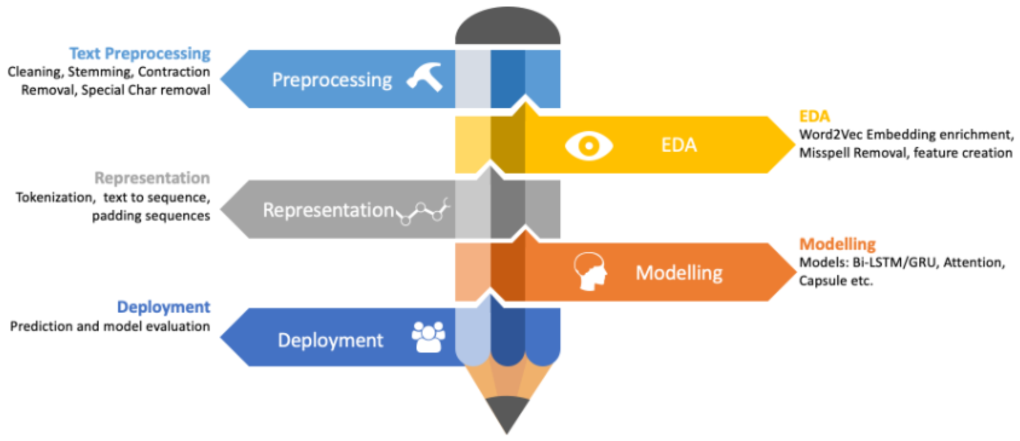
If None, no stop words will be used. max_df can be set to a value in the range [0.7, 1.0) to automatically detect and filter stop words based on intra corpus document frequency of terms.

Feature Extraction:

Feature extraction from text is done to feed as input to a machine learning model.

The questions in the dataset after preprocessing is considered as corpus for feature extraction. We used CountVectorizer() from ski-kit learn. It creates a matrix of numbers to represent our questions. CountVectorizer() takes what's called the Bag of Words approach. Each message is separated into tokens and the number of times each token occurs in a question is counted. CountVectorizer from sklearn is instantiated as an object. We will use fit() and then transform(), where using the fit method, our CountVectorizer() will learn what tokens are being used in our questions.

```
count_vectorizer = CountVectorizer(min_df=1, analyzer='word',
                                   tokenizer=word_tokenize,
                                   token_pattern=r'(\w{1,}),',
                                   ngram_range=(1, 5), stop_words = 'english')
count_vectorizer.fit(X_train)
train_vector =  count_vectorizer.transform(X_train)
test_vector = count_vectorizer.transform(X_test)
```


Text Preprocessing
Cleaning, Stemming, Contraction Removal, Special Char removal
Preprocessing
EDA
EDA
Word2Vec Embedding enrichment, Misspell Removal, feature creation
Representation
Tokenization, text to sequence, padding sequences
Representation
Modelling
Modelling
Models: Bi-LSTM/GRU, Attention, Capsule etc.
Deployment
Prediction and model evaluation
Deployment

Finally applying the processed data to the below Machine Learning Algorithms and compare the performance of both models.

Attention: Add attention to our neural network and propose an Attention-based Long Short-Term Memory Network for aspect-level sentiment classification. The attention mechanism can concentrate on different parts of a sentence when different aspects are taken as input.

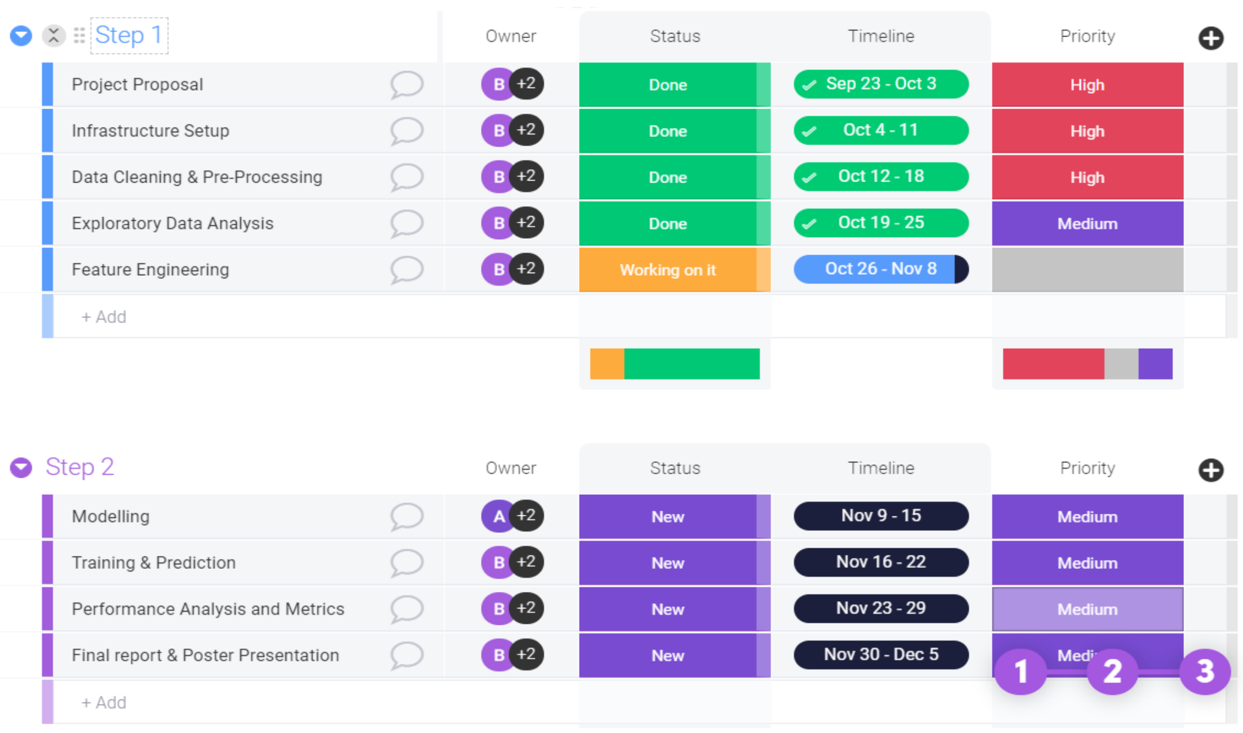
CNN: Use convolutional neural networks can be used as a recurrent structure to capture contextual information as far as possible, which may introduce considerably less noise compared to traditional window based neural networks

Notable Difference:

1. Data pre-processing and feature extraction is key in sentiment analysis and we are using the combination of Tokenization, N-Grams, Count vectorizer, Word embeddings which was not performed earlier.
2. Selecting the features in an iterative process and deciding on the best suitable features which gives better performance.
3. Applying the algorithms related to Attention & CNN to our processed model.

Updated Time table and Work Assignment:

Member Name	Responsibilities
Mahanth Mukesh (801034945)	1. Project Report 2. Infra Setup 3. Data Cleaning & Pre-processing 4. Modelling 5. Training & Prediction 6. Performance Analysis
Bharadwaj (801151165)	1. Project Report 2. Data Cleaning & Pre-processing 3. Feature Engineering 4. Modelling 5. Performance Analysis 6. Poster
Akhil Morampudi (801138186)	1. Project Report 2. Infra Setup 3. Data Cleaning & Preprocessing 4. Feature Engineering 5. Modelling 6. Training & Prediction



Step 1	Owner	Status	Timeline	Priority
Project Proposal	B +2	Done	✓ Sep 23 - Oct 3	High
Infrastructure Setup	B +2	Done	✓ Oct 4 - 11	High
Data Cleaning & Pre-Processing	B +2	Done	✓ Oct 12 - 18	High
Exploratory Data Analysis	B +2	Done	✓ Oct 19 - 25	Medium
Feature Engineering	B +2	Working on it	Oct 26 - Nov 8	
+ Add				

Step 2	Owner	Status	Timeline	Priority
Modelling	A +2	New	Nov 9 - 15	Medium
Training & Prediction	B +2	New	Nov 16 - 22	Medium
Performance Analysis and Metrics	B +2	New	Nov 23 - 29	Medium
Final report & Poster Presentation	B +2	New	Nov 30 - Dec 5	Medi
+ Add				

Reflections:

1. The link for the dataset in the project proposal document was missing which we had included in this report.
2. Literature Survey was missing in the initial report which was included in this report and the pros, cons of the existing approaches have also been mentioned and are proposed approach to address the existing challenges has been mentioned.
3. Proper Data Preprocessing and Feature Extraction, Selection are key in Sentiment Analysis, so in our approach we are using Word embedding's, Count vectorizer and N-Grams for text preprocessing which have better performance metrics over conventional techniques used for Text Pre-Processing and Sentimental Analysis.
4. Detailed plan of action and the updated task assignments has been updated and is kept in a readable format which was a proposal kept in the last proposal report.
5. Notable Difference has been updated.
6. Citing references has been updated.

References:

[1] C. Liu, Y. Sheng, Z. Wei, and Y. Yang, “Research of text classification based on improved tf-idf algorithm,” in 2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE), Aug 2018, pp. 218–222.

[2] O. Aborisade and M. Anwar, “Classification for authorship of tweets by comparing logistic regression and naive bayes classifiers,” in 2018 IEEE International Conference on Information Reuse and Integration (IRI), July 2018, pp. 269–276.

[3] Mungekar, Akshay, et al. "Quora Insincere Questions Classification."

[4] https://towardsdatascience.com/a-gentle-introduction-to-natural-language-processing- e716ed3c0863

[5] https://towardsdatascience.com/quora-insincere-questions-classification-d5a655370c47

[6] Quora Insincere Questions Classification.‖ [Online]. Available: https://kaggle.com/c/quora- insincere-questions-classification. [Accessed: 29-Jul-2019]

[7] Boiy, Erik, and Marie-Francine Moens. "A machine learning approach to sentiment analysis in multilingual Web texts." *Information retrieval* 12.5 (2009): 526-558.

[8] Yoon Kim. 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014).

[9] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the 2015 conference on empirical methods in natural language processing. 1422–1432.

[10] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A C-LSTM neural network for text classification. arXiv preprint arXiv:1511.08630 (2015).

[11] Gabbard, Samuel, Jinrui Yang, and Jingshi Liu. "Quora Insincere Question Classification."