Toxic Comment Classification

Abhishek Patil Rajkumar Pillai Hanmant Lokare

ABSTRACT

Cognitive computing is a term widely used to computer science world and refer to using a computer models to simulate human thought process. The learning of this models are then used to solve or answer uncertain questions. In this project, we are solving a a question of identifying the toxic comments on Wikipedia. Wikipedia allows user to comment on article or web page published on internet. And people can become abusive and use unethical words which might hurt other people's feelings. Wikipedia is considered to be a rich source of information from fields like science, art, literature, history and various others. Unethical and abusive comments can create negative impressions on various communities. This project focuses on classifying the abusive comments by using cognitive computing technique and machine learning (ML) approaches. We have followed a well known methodology from data mining called CRISP-DM to better understand the data and the process. We can performed various classifiers from ML and shared the performance analysis results.

1. INTRODUCTION

Cognitive computing is a fusion of cognitive science and computer science field like Artificial Intelligence. Cognitive science consist of learning how human brain works and simulate all its behaviour in machine. Human brain is capable of doing many complex work like interpreting emotion, learning languages, interpreting patterns by reading text, etc. Natural Language Processing(NLP) is one of the most studied and famous sub-field of cognitive computing. NLP deals with processing and analyzing large textual data-set and it generally consist of computer and human interaction.

Wikipedia is one the largest multilingual online encyclopedia. It consist of webpages which can be edited by anyone. There are many cases where people can be abusive and can behave unethical while commenting on particular topic. A research team founded by Jigsaw and Google is working on enhancing the online comments, conversation or any textual input provided by public. The improvement consist of removing abusive, vulgar or offensive text from that big corpus.

The dataset we are using is from Wikipedia detox project which was published by [1] under Wikipedia Talk Corpus on FigShare website. The data is gathered from Wikipedia comments and each of the comments are rated by humans and in this report we will call them as worker. Each worker is given a task to annotate the comment as toxic or not. We initiate the task by applying certain pre-processing steps including removing stop words and converting words to lowercase. Since this is a classification problem, we are going to approach the task by applying Machine Learning techniques including Multinomial Naive Bayes Classifier to classify if a comment is toxic or not.

The rest of the paper will consist of following sections: Section 2 discuss the related work done by researchers for solving the toxic comments problem. Section 3 consist of discussion on methodology followed in this project, Section 4 consist of implementation and section 5 discuss the results of the models. Section 6 will explain all lessons we learned during this project. Section 7 is dedicated for the future scope of this project and last section 8 present the conclusion of the project.

2. RELATED WORK

Many researchers are interested in solving Wikipedia toxic comments problem. In [2], the author's discuss the idea of combining various models to create an ensemble that performs better than individual models. They also further discuss the challenges with existing state-of-the-art methods. One of the main challenges researchers faced is over increasing data on daily basis. In [3], the author's discuss deep learning approaches like Convolutional Neural Networks (CNN) and cloud computing to solve the big data problem occurred in Classification task of toxic comments in Wikipedia. Similarly, in [4] the author's discussed a novel approach of using Apache Spark with several word embedding technique to solve toxic comment problem.

3. METHODOLOGY

Every data mining or data related project is successfully documented when audience can know the answers of few key questions like: What is the aim behind it? Why did you do a particular step? What was the process? What are the results?. The key questions can be answered by following a well known methodology called CRISP-DM (Cross Industry Process for Data Mining). CRISP-DM is considered to

be robust and common approaches used by data mining experts. This project is well suited to use this methodology as it involved data analysis, data preparation and data classification. Figure 1 shows the steps involved in CRISP-DM model. Following are the iterative steps from CRISP-DM.

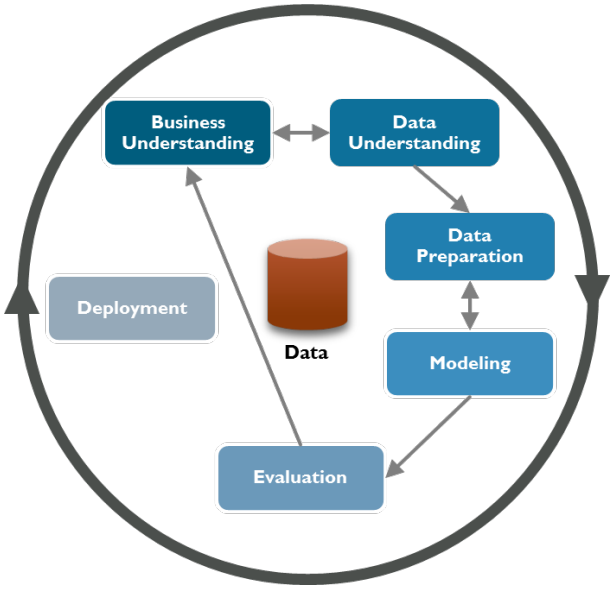


Figure 1: CRISP-DM Model

1. Business Understanding: Before starting any data mining project, it is important to understand the goals of the projects. This step involves understanding the goals and objective behind the project and considered to be an important step in this methodology.
2. Data Understanding: This step involves understanding and defining important features of the data. It is also an important step in the methodology as it give an overall idea about the data involved in the project.
3. Data Preparation: In this step we will clean and prepare the data in order to better fit for further steps and analysis. It consists of steps like identifying and correcting inaccurate data records from databases, tables or files.
4. Modeling: Modeling step consist of deciding which ML pr cognitive computing technique are better fit for our data and main goals. It also involves designing, building and analysing the models in order to achieve better results.
5. Evaluation: Evaluation step involves analysing the the results from modeling step, summarizing and comparing the results of the models. This also involves documenting the results for further development.
6. Deployment: This is the final step in CRISP-DM methodology which consist of deployment of best models selected to solve the project problem. In this project we will present the best models which can identify the toxic comments on Wikipedia which will potentially help to keep track of those people and remove the comments too.

4. IMPLEMENTATION

4.1 Business Understanding

The main aim of the project is identifying the toxic comments on Wikipedia which are unacceptable and harm the community by loosing interest in contributing the knowledge. We are using machine learning and cognitive computing algorithms and technique which will help the developers to keep track of the toxic comments and people associated with it. The models built in this project can be a part of bigger models which in combination can narrow down the scope of toxicity required to find. In this project we will also aim to clean as much as possible unwanted data that can help our model to better learn and give best results as possible. The models can be used to remove abusive or offensive texts which will promote healthy discussions in online community.

4.2 Data Understanding

The dataset consist of three files toxicity_annotations.tsv, toxicity_annotated_comments.tsv, toxicity_worker_demographics.tsv. The dataset consist of total more than 100k labeled discussion comments form wikipedia pages in English.

Column Name	Description
rev_id	Revision ID made by MediaWiki of the comment
comment	Actual comment text made by user on the talk page which is annoted by MediaWiki parser
year	Year on which comment is posted
logged_in	Flag 1 if the user commented is logged in otherwise 0
ns	NameSpace of the page
sample	Flag random if the comment came from random sampling of the comments otherwise blocked
split	For building the model, the comments are split into train, dev and test sets.

Figure 2: Description of file toxicity_annotated_comments.tsv

Column Name	Description
rev_id	Revision ID made by MediaWiki of the comment
worker_id	Random crowd-worker ID
toxicity	Flag 1 if the worker consider comment toxic otherwise 0
toxicity_score	Flag -2 if the comment is very toxic, 0 if neutral and 2 if healthy

Figure 3: Description of file toxicity_annotations.tsv

Fig 1,2,3 consist of column names and the description of each column in the dataset.

4.3 Data Preparation

File toxicity_annotations.tsv contains toxicity score of each comment manually annotated by multiple crowd-workers.

Column Name	Description
worker_id	Worker ID
gender	Gender of the worker
english_first_language	Flag 0 if the worker's first language is not english otherwise 1
age_group	Range of the workers
education	Highest education of the worker in degrees achieved

Figure 4: Description of file toxicity_worker_ demographics.tsv

Since each comment has multiple scores, we take the mean value of all toxicity scores for that comment and assign True if the mean is above 0.5 or else we assign False. These boolean values are stored in a new column named 'is_toxic'. The original data is split into train and test data sets using the column named 'split' in Figure 1. As a part of preprocessing, we convert the comments to lower case, remove stop words(for English language) and create word count vectors for every comment using CountVectorizer from Scikit-Learn. The word count vector is a vector that stores the frequency of the word in each comment for all the unique words in the entire dataset where each row represents a sentence and each column represents the word.

5. MODELING OVERVIEW

For creating a classification model we have followed a series of steps as shown in figure 5:

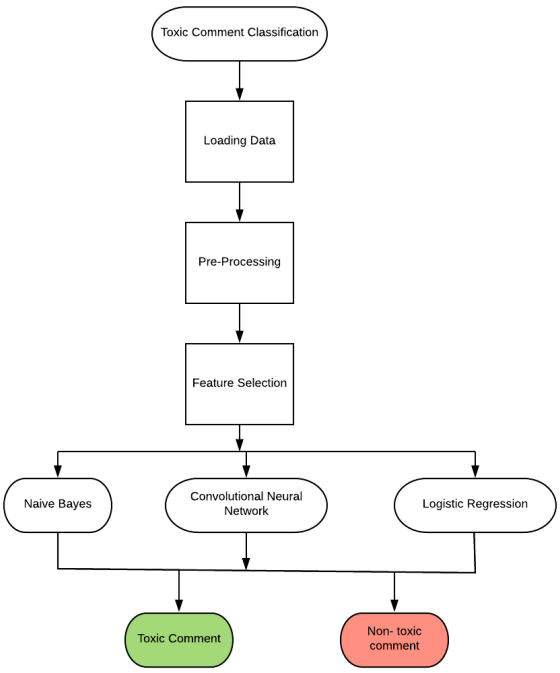
1. We have load the data data from the all the CSV file to our program.
2. We preformed multiple steps to do pre-processing steps which are explained in section 4.3.
3. As the features are an important part of any ML algorithm because it determine what our model learns, we have selected important features with respect to model selected. The features selected are mentioned in individual modeling step.
4. Furthermore, we have design and implemented models based on best features selected from step 3.
5. At last, we have perform and discussed performance of each model in section 5.2

5.1 Modeling

5.1.1 Naive Bayes Model

Multinomial Naive Bayes (NB) model is used to determine the probability of the comment to find out if the comment falls under toxic category or not. This model is based on the Bayes rule which assumes that there is a strong naive independence amongst the features and that all the attributes are conditionally statistically independent. Multinomial NB is an optimization of Naive Bayes classifier which calculates the probabilities by keeping track of the number of occurrences of the word and uses a bag of words to identify a comment as toxic or non-toxic. For training the classifier, the frequencies of all the unique words in the documents are used as features to the model.

Visualizing the data, we observed that there exist particular words in the dataset, where these words have a special specific probability of occurring in the toxic comments or



System Design

in the non-toxic comments. For example, the word "hell" was observed only in toxic comments while the word "nice" occurred in most of the non-toxic comments. For such specific words, the Bayesian networks were found to give a high probability if they were toxic and a low probability if they were non-toxic. Figure below represents the formula for calculating the probability of a comment being toxic :

$$P(\text{Toxic} \mid \text{word}) = \frac{P(\text{Toxic}).\,P(\text{word} \mid \text{Toxic})}{P(\text{Toxic}).\,P(\text{word} \mid \text{Toxic}) + P(\text{non-Toxic}).\,P(\text{word} \mid \text{non-Toxic})}$$

Where,

P(Toxic | word) = Probability of a comment being toxic given a word
P(Toxic) = Probability that any word is toxic
P(word | Toxic) = Probability of a word occurring in a toxic comment
P(non-Toxic) = Probability that any word is non-toxic
P(word | non-Toxic) = Probability of a word occurring in a non-toxic comment

Figure 5: Probability that a comment is toxic

The probability for each word in the comment is calculated using the equation in Figure 5 and overall probability of the comment being toxic is a simple multiplication of the probabilities of all the words in that comment based on the naive assumption of independence. Similarly, the overall probability of the comment being non-toxic is also obtained. The label '1' for toxic or '0' for non-toxic is given based on the maximum of their corresponding probabilities.

5.1.2 Convolutional neural network model

Convolutional Neural Networks (CNN) consist of multiple layers of Neural network architecture that are used for classification task. The CNN are used in numerous image

classification tasks due to its ability to understand the properties of neighboring pixels in an image. Similarly, in text classification, CNN can understand the properties of neighboring words in a sentence.

The convolutional layer of CNN perform convolution on the input and generate a matrix of features as output and a bias is added to this output. The learning process of CNN involves changing of this weights and biases in the output to determine the strong neuron connection between neuron in current layer and neuron in next layer. The Backpropagation algorithm is the core of the training process. The algorithm requires a vector of outputs and a vector of inputs in each iteration of the training process. These input and output vectors are compared to determine the training error which is also called as value of loss function. The main aim of the training process to reduce the value of cost function as much as possible to increase the accuracy of the network. Ideally, a network must be trained until the loss function does not reach the minimum possible value to avoid underfitting. The model must not be trained for next iteration if the value of loss function is increasing for two consecutive iterations to avoid overfitting. For this purpose, the tolerance level was set to 0.00000001 which means any difference of value in loss function for two consecutive epochs beyond this tolerance level will cause the model to overfit, so the training of CNN must be stopped.

The MLP classifier available in the sklearn package was implemented as CNN. The CNN model was intended to classify whether a comment is toxic or non-toxic. It consist of hidden layers of size 1,2,3,5. The 'relu' activation function was used to activate the neurons in the hidden layers. The network was trained for 30 iterations. The optimizer used was stochastic gradient descent and cross-entropy loss criterion function and 0.1 as learning rate. The network trained itself to classify the dataset, and it ended up classifying the test dataset at about 78 % accuracy. Below is the confusion matrix of the test dataset.

After training the initial model of CNN, further efforts were taken to improve the accuracy of the model by chaining the optimizer, activation function, decreasing the learning rate, Increasing the number of hidden layers and increasing the number of iterations to observe if it increases the accuracy of the network. All of the above combination of techniques failed to improve the accuracy of network beyond 78% but increased the time for training the CNN. The classification report of CNN is provided in Section 5.

5.1.3 *Logistic regression model*

Logistic regression is most often used when the target variable is categorical like toxic or not toxic in this toxic comment classification dataset. Unlike linear regression, in which there is threshold value which serves as a reference point to classify the labels depending the position of the value above or below the threshold value, there is no need of calculating a threshold value in logistic regression. In data pre-processing stage, tf-idf count vectorizer was used in union with character vectorizer which converts which computes the char level features in the document followed by tf-idf transformer. The logistic regression used for this dataset is binary logistic regression which predicts a comment as toxic or not toxic. The gradient descent technique is used to minimize the cost function and the solver used in this project is SAGA which is a variation of Stochastic average gradient descent. The output of logistic regression is actually an estimated probability showing the chances of a predicted label for a comment being toxic or not. The value of this estimated probability is compared with sigmoid function where the hypothesis is that if output approaches the infinity value then the predicted label is toxic and if the output approaches the negative infinity then the predicted label is not toxic.

The benefit of using a logistic regression over linear regression is generating a decision boundary is possible for the datapoints by visualizing the distribution of points belonging to each target variable in the feature space. The classification report of Logistic Regression is provided in Section 5.

5.2 Evaluation

5.2.1 *Evaluation Metric*

The following evaluation metric is used in order to evaluate the performance of both the algorithms on the Wikipedia comments data set.

Evaluation Measure	Evaluation Function
Accuracy	$Acc = \frac{TN+TP}{TP+FN+FP+TN}$
Recall	$r = \frac{TP}{TP+FN}$
Precision	$P = \frac{TP}{TP+FP}$
F-measure	$F = \frac{2pr}{p+r}$

Where,

Accuracy: Percentage of correctly identified toxic and non-toxic comments
Recall: Actual toxic comments actually retrieved
Precision: Percentage of relevant toxic comments
F-measure: Weighted average of precision and recall
True Positive (TP): Number of correctly classified toxic comments
True Negative (TN): Number of correctly classified non-toxic comments
False Positive (FP): Number of misclassified non-toxic comments
False Negative (FN): Number of misclassified toxic comments

Figure 6: Evaluation measures for toxic comments

5.2.2 *Evaluation of different approaches*

Evaluation Measure	Naive Bayes	Convolutional Neural Network	Logistic Regression
Accuracy	0.7823	0.78	0.82
Recall	0.4354	0.43	0.39
Precision	0.4965	0.45	0.48
F-measure	0.4639	0.48	0.63

Figure 7: Evaluation of different approaches

6. RESULTS

The results of the experiments are present in figure 7 and figure 8.The accuracy of Multinomial Naive Bayes, CNN and Logistic Regression model model are 78%,78% and 82% respectively but the precision and recall of these models are in the range 0.39 to 0.49 which shows that these models are not sure about the predictions it has made. So, these models may not be the best model to classify if a comment is toxic or not. But the precision ,recall and accuracy of the models

can improve if training samples are more so that models will be able to learn more features in comment to determine is it is toxic or not.

Algorithm	Accuracy
Naive Bayes	0.782
Convolutional Neural Network	0.78
Logistic Regression	0.82
Decision Tree	0.7675
Adaboost	0.81
K-Nearest Neighbour	0.80
Random Forest	0.798

Figure 8: Comparative Analysis of different approaches

7. LESSON LEARNED

First of all, we learnt how to make use of unique words and their frequency as features as the features for the model. We found that there are some specific words in the data which have higher weights as compared to other words in classifying into toxic and non-toxic. We learnt how data pre-processing is an important part for creating a better model how Lemmatization is used to group different kinds of words together as a single word. Wordnet is a functional library that is used to perform Lemmatization in our work. Bringing the word to is original form (stemming) decreases the overall number of unique words and used to reduce the dimentionality of the model. Overall, Such several data cleaning techniques were learnt and implemented in order to achieve greater accuracy for the classifier.

8. CONCLUSION AND FUTURE WORK

This paper presents different approaches for finding toxic comments using classification algorithms which may pose a threat of harassment to people and might make them express less as some of the comments are quite abusive. In order to tackle this threat, Naive bayes and CNN based classifiers are used to correctly classify the data in to toxic and non-toxic. The results of both the approaches are discussed in the evaluation and the results section. Several data cleaning techniques including stemming and lemmatizing using WordNet have proved to reduce the dimentionality (complexity) as well as the overall accuracy of the model.

Logistic regression is found to give the best accuracy on the toxic comment wikipedia dataset and the reason is because other approach like Naive Bayes assumes that the all the features are conditionally independent and has a high bias but low variance. Therefore, Naive Bayes performs the best when the data is found to be biased.

For future work, better data cleaning techniques must be arranged in order to reduce the complexity and improve accuracy as current techniques does not work the best on neural network models like CNN. More tuning of the parameters must be performed to find out which are favourable for our learning models. Other than that, our current models have some limitations such as it is unable to detect sarcasms. In future, more flexible and vigorous system can be developed that identifies sarcastically toxic comments.

9. REFERENCES

[1] Nithum Thain, Lucas Dixon, and Ellery Wulczyn. Wikipedia talk labels: Toxicity, Feb 2017.

[2] Betty van Aken, Julian Risch, Ralf Krestel, and Alexander. Challenges for toxic comment classification: An in-depth error analysis, 2018.

[3] Spiros V. Georgakopoulos, Sotiris K. Tasoulis, Aristidis G. Vrahatis, and Vassilis P. Plagianakos. Convolutional neural networks for toxic comment classification. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, SETN '18, pages 35:1–35:6, New York, NY, USA, 2018. ACM.

[4] Salvatore Carta, Andrea Corriga, Riccardo Mulas, Diego Reforgiato Recupero, and Roberto Saia. A supervised multi-class multi-label word embeddings approach for toxic comment classification. In *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, Vienna, Austria*, pages 17–19, 2019.