TOXIC COMMENT CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK

Anoushka Samyal[1], Rahul Singh[2],Riddhi Shinde[3], Ms. Manya Gidwani[4]
[1]Student (B.E), [2]Student(B.E), [3]Student(B.E), [4]Project Guide
Department of Information Technology,Shah and Anchor Kutchhi Engineering College,University of Mumbai,Mumbai - 400088,India

Abstract:In today’s evolving world ofsocial media,the issue of trolling and online abuse has become increasingly prevalent. Hence, it is necessary to police the toxic comments which showcase such behavior.Due to extreme online harassment and cyber bullying it has become imperative to work on models that help curb these problems. In this paper, we build an efficient model to identify and categorize toxic comments using deep learning algorithms. Deep Learning methods are starting to out-compete statistical methods on some challenging NLP problems with singular and simple models.We worked on a public dataset which is a corpus of Wikipedia comments available on Kaggle. In this work, we used word2vec + Convolutional Neural Network (CNN) approach. We tested two models, CNN with word embeddings and CNN with character embeddings. Sentimental Analysis is the field that studies and analyzes people’s responses and acceptance towards entity using text analysis computational and algorithms to help to determine people’s textual reactions if they are toxic, severe toxic, threat, identity hate, obscene or insult. We trained our dataset using keras library. We show that our CNN with word embeddings model performed with an AUC of 0.98.Here we show that CNN with word embeddings reaches the highest performance.

Index words:Convolutional Neural Networks, Deep Learning, Natural Language Processing, Sentiment Analysis, word2vec.

I. INTRODUCTION

As the online forums and discussions have increased day by day, there is tons of text information emerging out of these forums. Sentiment Analysis is one of the important factors as it deals with text information and has a wide variety of applications worldwide. It helps the computer process opinions expressed in a piece of text using natural language processing. In the past, sentiment analysis was done using traditional models like Naive Bayes (NB) and support vector machines (SVMs).

Recently, researchers have been using deep learning instead of other traditional models to achieve better results in sentiment analysis. Deep learning is a set of techniques that help you to parameterize deep neural networks with many layers and parameters. It is a subset of machine learning. There are many different types of deep learning models; one of the most efficient and sought-after model is Convolutional Neural Network (CNN). CNN or ConvNet resembles a class of deep neural networks, most commonly applied to analyzing visual imagery, speech-recognition, object detection and other areas. These works have proved that CNN can give great results in sentiment analysis as well.CNN is a powerful deep model used widely for classification problems. Huge advantage of using CNN is that it reduces the amount of work by labeling the whole sentence artificially unlike RNN where labels of each word or group of words are required. Now, the challenge is modelling the data in a way which can be accepted by the CNN. Hence, we use word2vec since it helps to classify large entities and translates words to vectors which can be easily understood by CNN. There are also many different methodologies like B-O-W (Bag of words) model which follow the traditional encoding method.

It is an alternative way of extracting features from text that can be used for classification. In Bag of words model, the representation of text describes the occurrence of words within a document and for this purpose it only involves a vocabulary of known words and their corresponding measure of the presence. Although theoretically simple and practically efficient this model involves several technical challenges like the first step is to build the Document-Term-Matrix (DTM) from input documents. This is prepared by vectorizing documents creating a map from words to a vector space.

Traditional bag-of-words model (BOW):

1. Scalability challenges: Bag of words model is used to encode every word that is present in the vocabulary as one hot encoded vector i.e. for a vocabulary size |R|, each word is represented by a |R| sparse vector with 0 at every other index and 1 at index corresponding to the word.
2. No respect to semantics of words: However the words “mobile” and “smart phones” are often used in same context but in bag-of-words model the vectors of these words are represented as orthogonal vectors hence the problem becomes serious while modelling the sentences.
3. No respect to order of words): In BOW model, the sentences “this is bad” and “is this bad” have similar vector representation which is one of the disadvantages of BOW model.

Due to these challenges we found that word2vec is a better suitable technique for working on our corpus. The remainder of this paper reviews the related work on toxic comment classification and shows the proposed architecture. Experimental results and analysis are presented as well and finally, we make some conclusion.

II. RELATED WORK

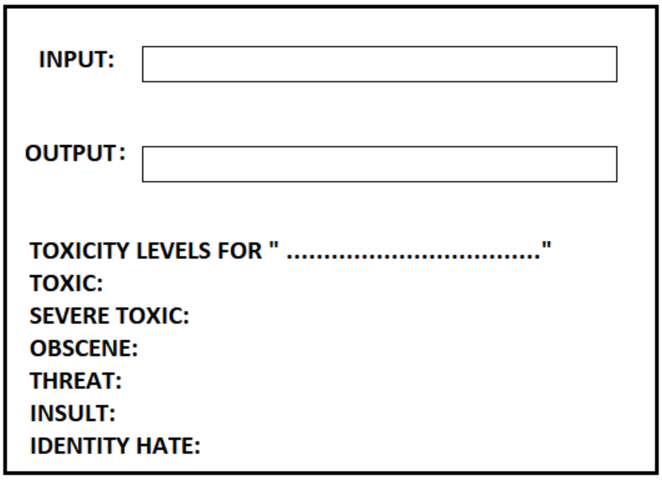
A. Google's Perspective Tool

Google's perspective tool is an API built by Google and jigsaw to tackle the toxic comment problem rising today in the online forum. Jigsaw is a technology incubator created by Google and in February 2017, they rolled out the perspective tool to maintain decorum over online forum. However, to gain stability and improve the online technology environment, Google created perspective tool to minimize the effect of toxic comments and support online discussions. The technology team of Google and jigsaw came up with this tool to help spot toxicity online. There can be difficulties while discussing things you care about. Hence, many people give up on finding different opportunities and opinions and after a while stop expressing. Hence, to encourage people participation over online conversations both parent and the incubator came up with their tool to pacify the online environment.

This API makes it feasible and easy to host online discussions. Since, it uses machine learning model to score the perceived effect a sentence might have on a discussion. Google and jigsaw created this experiment using perspective to illustrate when comments might be perceived as "toxic" by people. The online tool is available at the following website: https://www.perspectiveapi.com/#/

In this paper, we build a model where our API will recognize the toxic comments and will distinguish the toxicity into six different categories as:

a) Toxic
b) SevereToxic
c) Obscene
d) Threat
e) Insult
f) Identityhate



INPUT:

OUTPUT:

TOXICITY LEVELS FOR " ...."
TOXIC:
SEVERE TOXIC:
OBSCENE:
THREAT:
INSULT:
IDENTITY HATE:

Fig.1: API to determine toxicity levels for toxic comments and its classification.

Severetoxic: poisonous capable of causing serious injury or death toxic effect.
Toxic: very unpleasant or unacceptable capable of causing a lot of harm.
Obscene: Very offensive or rude capable of spoiling one's senses.
Threat: warning of killing or punishing one for what he/she wants.
Insult: treated is respectfully or scornful abuse.
Identity hate: practice harsh hatred and hostility.

III. WORKING MODEL: WORD2VEC AND CNN

We want to use a deep learning model to solve the text sentiment classification problem. Pre-training techniques play a vital role in helping deep learning algorithms accept data and get higher accuracy. In this approach we first used the word2vec technique to translate natural language into mathematical vectors which could be understood by the deep learning model. As we want to classify our sentences using CNN, we need the words input as vectors. CNN cannot understand the natural language like humans. Hence, here word2vec is a key-enabling factor for the success of CNN in dealing with non-image data.[1] Thus, in this paper, we use a framework of Word2vec + CNN. Firstly, we use the word2vec to transform the words to the vectors, so that we can build up the sentences' vectors. Secondly, in order to classify the sentences to different sentiment labels, we input the sentences' vectors to CNN. Our dataset(corpus) consists of the Wikipedia comments which are pre-processed using natural language processing technique

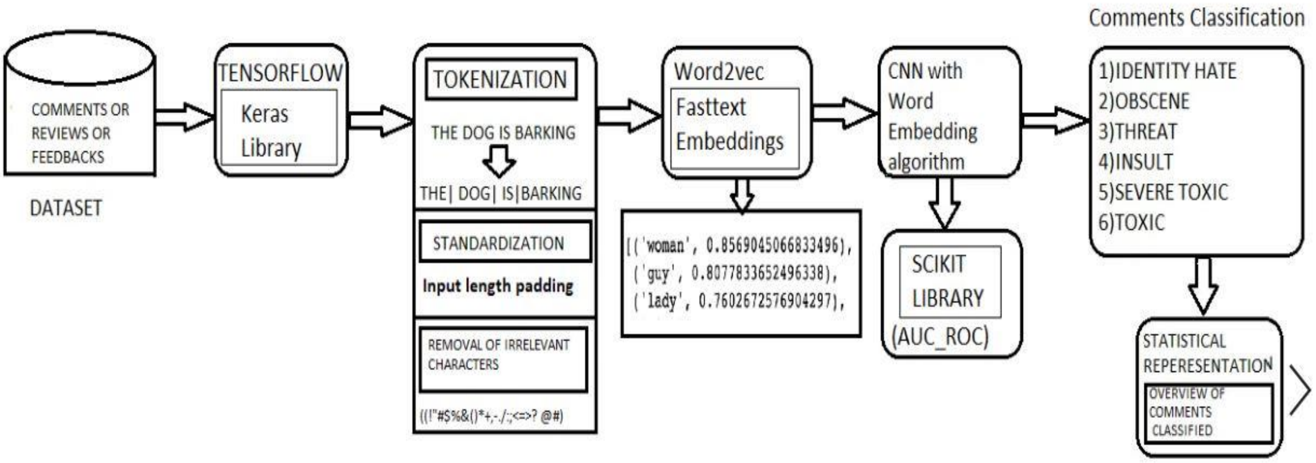


Fig.2: Working model

A. Pre-processing

In this paper, we are preprocessing our data so that we can input it into our neural network with a series of steps:

a) Removal of irrelevant characters (!"#$%*+,-./:;<=>?@[\\]^_\)
b) Converting to lower case letters (HEY->hey)
c) Tokenization of words (how are you= [1,5,13])
d) Standardization of input length with padding (how are you= [1,5,13,0,0,0])

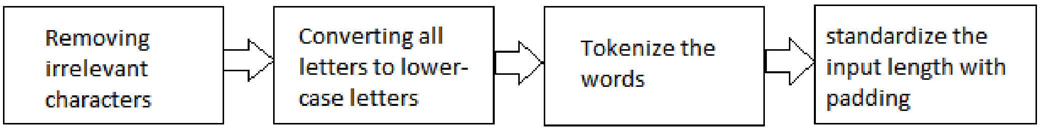

Removing irrelevant characters
Converting all letters to lower-case letters
Tokenize the words
standardize the input length with padding

Fig.3: Steps for Pre-processing the data.

As we know certain real-world data have errors, incomplete, inconsistent, raw data that needs to be transformed in understandable format. Hence, data pre-processing is an improvising method for resolving issues and unequal distribution of data across datasets. However, a clean dataset is always preferred as perfect dataset for better benchmarks and better performances. The major steps for pre-processing include data cleaning, data transformation and data reduction. The below given example explains how tokenization is done after it is embedded into a tokenizer. We input two sentences into tokenizer and obtain a sequence of tokenized text. Raw data contains noisy data, inconsistent data, presence of outliers, uncertain behavior of data etc. Hence, to get meaningful information we need to pre-process data.

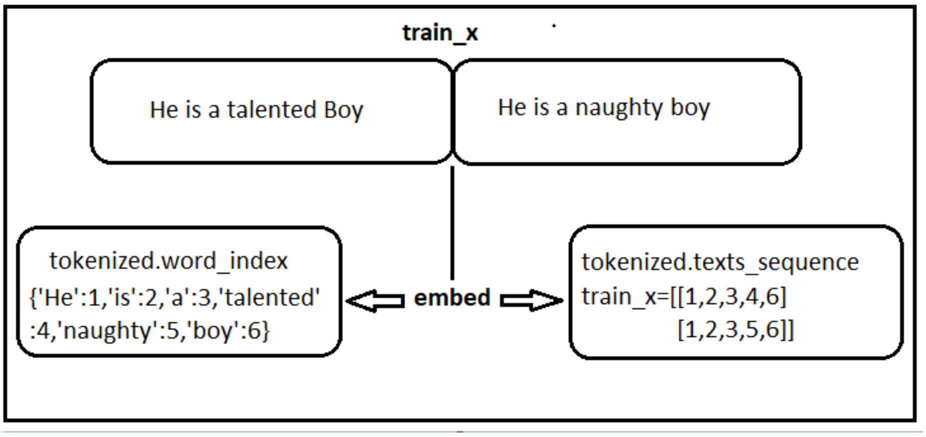


Fig.4: Example for tokenization of data

B. Word2vec

A person expresses an idea by using words, signs etc. This idea that is expressed in the work of art, writing How does it relates to the computer? The answer lies in the Word Net that uses taxonomy that contains two things: hypernyms relationships and synonym set.

We all might have read about discrete representations. But the only problem with the discrete representations is the missing nuances, e.g., synonyms adapt, expert, practiced, proficient, skillful? Also, the missing new words (impossible to keep it up to date): wicked, badass, ninja, whats'up and how's that nifty. Hence this proves to be subjective and even requires human labor to create and also adapt. Since we all know it is also hard to compute accurate word similarity. Hence there arise a lot of incompleteness. Therefore, it is quite unclear because of the unavailability

From symbolic to distributed representations

The problem, e.g., for web search
1 .If user searches for [Apple notebook battery size], we would like to match documents with "Apple notebook battery capacity.
2 . If user searches for [cheap motel near me], we would like to match documents with "cheap hotel near me"
But
Motel[000000000010000] raised to t
hotel [000000010000000] isequals to 0

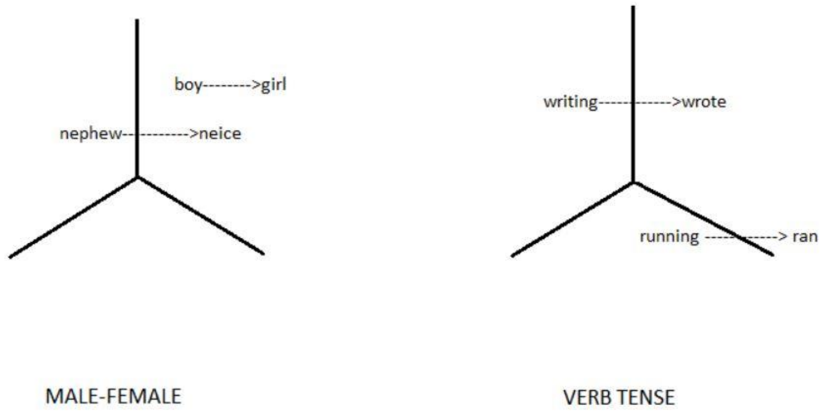


Fig 5: Relationships of words in word embeddings in three dimensional.

The intuitive understanding can be understood by the following word to vector analysis.

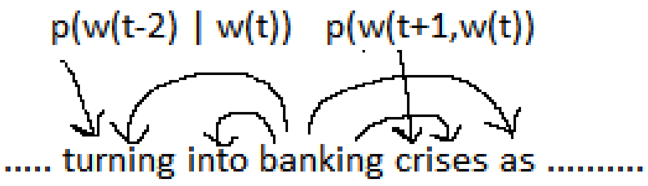


Fig.6 : Example of word2vec

Objective function: Maximize the probability of any context word given the current center word:

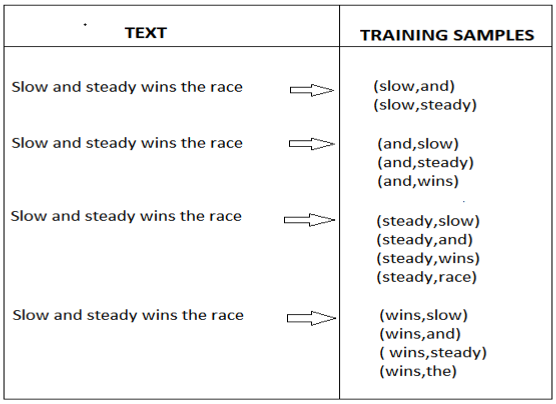
- Terminology: Lossfunction(LF) = costfunction(CF) = objective function(OF)
- Usual loss for probability distribution: cross – entropy loss
- With one-hot w (t + j) target, the only term is the negative probability of the true class

Main idea of word2vec is to Predict between every word and its context word. However, there are two algorithms named skip grams (SG) and continuous bag of words (CBOW) model.

SG: predict context words given target (independent position)

CBOW: predict target word from bag-of-words context.

Given a set of sentences (CORPUS) the model loops on the words of each sentence and either tries to use the current word of to predict its neighbors (its context), in which case the method is called "Skip-Gram". When it uses each of these contexts to predict the current word, in such case the method is called "Continuous Bag of Words" (CBOW).



TEXT	TRAINING SAMPLES
Slow and steady wins the race ⇨	(slow,and) (slow,steady)
Slow and steady wins the race ⇨	(and,slow) (and,steady) (and,wins)
Slow and steady wins the race ⇨	(steady,slow) (steady,and) (steady,wins) (steady,race)
Slow and steady wins the race ⇨	(wins,slow) (wins,and) (wins,steady) (wins,the)

Fig.7: Traditional Skip Gram method.

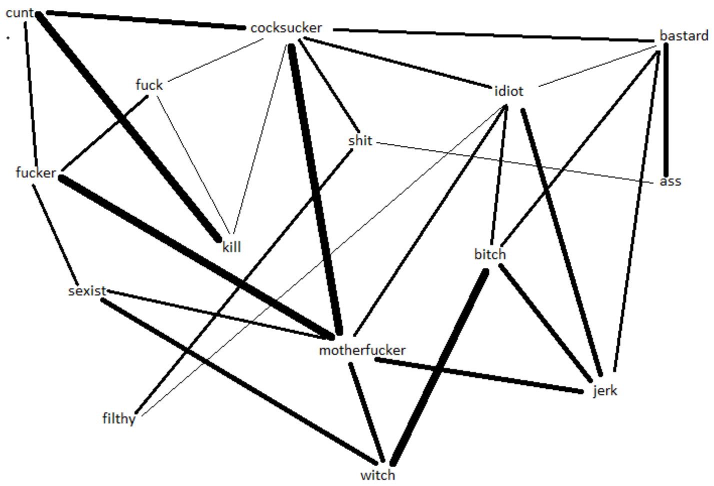


Fig.8: A diffused network for the words to represent the relationship in the Wikipedia dataset of comments. The nodes represent the words in the dataset. The width represents the strength of the word similarity.

C. CNN

Neural networks make assumptions by studying the relationship between patterns of your data and some observed response. In Convolutional Neural Networks (CNN) every network layer is a detection filter for the presence of specific factors known as features or patterns present in the original data. The first few layers in a CNN detect (large) features that can be recognized and compared relatively easy. Later layers detect smaller features that are more abstract. The final layer of the CNN is able to make specialized classification by integrating all the specific patterns detected by the preceding layers in the input data. Convolutional Neural Networks are multistage trainable Neural Networks architectures developed for tasks of classification. These stages consist of each of the following layers described below:

(1) Convolutional Layers: They are major components of the CNNs. A Convolutional layer consists of a number of matrices of kernel wherein convolution is performed on their input and an output matrix is produced of features where a partial value is added. The learning procedures goal is to train the kernel weights.

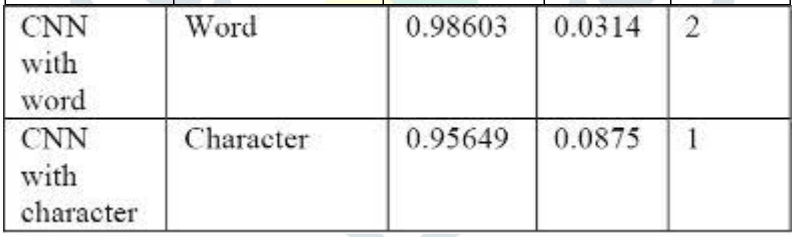
(2) Pooling Layers: They are also integral components of the CNNs. The pooling layer performs dimensionality reduction of the input feature images. The most common pooling function is the max-pooling function. Its function is to take the maximum value of the local neighbourhoods.

(3) Embedding Layer: It is a special component of the CNNs for text classification problems. The embedding layer transforms the text inputs into a suitable form for the CNN. Here, each word of a text document is modified into a dense vector of fixed size.

(4) Fully-Connected Layer: These are usually the last few layers of CNN. In these layers, the input would be in the shape designed by the earlier stages of CNN. They perform classification based on the input received from the previous CNN layers [3].

So to put it briefly, Convolution layer and subsampling layer perform feature extraction, whereas fully connected layers perform classification based on the features by previous layer.

We will train a Convolutional neural network after getting the pre-trained word vectors from word2vec technique. The CNN model we are going to use consists of convolutional layer in one dimension which will lay across the concatenated word embedding for each input comment. In total, the convolutional layer consists of 128 filters with a kernel having a size of five. Hence, each convolution will consider a window of five-word embeddings. The layer before the output layer is the input layer which is fully connected with 50 units.



Model	Embeddings	AUC	Loss	Epoch
CNN with word	Word	0.98603	0.0314	2
CNN with character	Character	0.95649	0.0875	1

Table 1: Comparison of CNN with word and CNN with character algorithm

IV. CONCLUSION

Toxic comment classification is an active research field with number of multi-label classification models. We know toxic sentences are online discussion terminators and therefore in this paper we build a framework called Word2vec + CNN with word embeddings to classify and categorize the toxic sentences and give the toxicity level for each sentence input. Our proposed model will help to identify the toxic comments which ruin the online (forum) conversations. We used word2vec and pre-processing techniques to pre-train the data and obtain its vector form that can be input into the neural network. However, we found that in our case the CNN with word embeddings proved slightly efficient and gave more AUC than CNN with character embeddings. We analyzed this when we passed a toxic sentence to our API and it categorized it into different toxicity levels. Since toxic comments may force someone to leave an online conversation or discussion which can be rude, threatful or disrespectful. We have built this model to maintain decorum and improve online discussions that can be inspiring for future researchers.

V. REFERENCES

[1] Ouyang Xi, Zhou Pan, HuaLi Cheng, LiuLijun (2015)-2015 IEEE International Conference on Computer and Information Technology; UbiquitousComputing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing,"*Sentiment Analysis Using Convolutional Neural Network*"

[2]Georgakopoulos Spiros V., TasoulisSotiris K., VrahatisAristidisG., PlagianakosVassilis P.(2018)- "*Convolutional Neural Networks for Toxic Comment Classification*"

[3] DCunhaMeven, BhangaleNeha(2018)- "*Toxic Comment Classification*"

[4] Yuling Chen, ZhiZhang(2018)- 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA),"*Research on text sentiment analysis based on CNNs and SVM*"'

[5]Lin Li, Linlong Xiao, Nanzhi Wang, Guocai Yang*, JianwuZhang(2017)-2017 3rd IEEE International Conference on Computer and Communications, "*Text Classification Method Based on Convolution Neural Network*"

[6]Dos Santos, CceroNogueira, and MairaGatti.(2014)- "*Deep Convolutional Neural Networks forSentiment Analysis of Short Texts.*" COLING.

[7] Pang B, Lee L(2008)- "Opinion mining and sentiment analysis," Foundations and Trends in Information Retrieval, 2.1-2: pp. 1-135.

[8]Kim Y.(2014)-" Convolutional neural networks for sentence classification", Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. 2014: 1746-1751.

[9] O. Abdel-Hamid et al.,(2014)- "Convolutional Neural Networks for Speech Recognition," IEEE/ACM Trans. Audio, Speech, Lang. Process., vol. 22, no. 10, pp. 1533–1545.

[10] Y. Kim(2014)- "Convolutional Neural Networks for Sentence Classification," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014).

[11] Theodora Chu, Jue Kylie, Max Wang-"Comment Abuse Classification with Deep Learning".

[12] Kim, Yoon(2014)- "Convolutional neural networks for sentence classification." arXiv preprintarXiv:1408.5882 .