DETOX
A toxic content classifier

Akanksha Mohan (amohan7), Rishi Rai (rrai2), Shravani Yellu (syellu), Tushar Dey (tdey)

1 Introduction and Background

1.1 Problem statement

In this project we are trying to decide whether comments on the internet are toxic or not. With increasing number of people coming online everyday, we are having more diverse and constructive conversations which is enabling us to better understand different views and beliefs. But with more people being connected online there are handful of bigoted people trying to suppress the conversations by spewing their racist and harmful believes. There is an inherent need for stopping these type of objectively vile comments which ceases the open and constructive conversations. In this project we are working toward identifying these harmful comments and classify them efficiently. The traditional machine learning approach like logistic regression, SVM etc. have limited capacity of understanding sentences with bag of words approach. Most of the traditional approach lacks in preserving context of the word which can lead to miss-classification of even constructive criticism as toxic. In recent decade, there has been numerous development in the field of NLP with the resurgence of deep learning. These deep learning models including deep Multi-layer perception, Convolutional Neural Network and Long Short term memory has time to time proven to perform excellently for various NLP tasks. Since the data-set we are using is quite skewed, we also wanted to test the robustness of these models against our data and compare that with that of the traditional models.

1.2 Related Work

Traditional approach follow the classical two stage strategy of extraction of handcrafted features followed by traditional machine learning classification model[12][13][14][15]. Das et al.[16] compares performance of bag of words with TF-IDF model and then applies on different traditional model and confirms that SVM performs better. In a prominent paper by Yoon Kim[1], the author implemented a single layer CNN model which was trained on top of a pre-learned word vectors for various tasks. In their research they found that even a very shallow model without much hyper-parameter tuning was able to outperform previous state of the art for 4 out of the 7 NLP task including sentence level classification. Since our problem domain involves the intricate task of classifying text as toxic or not and the data set involved is substantially large, we extend the idea to implement a comparatively deeper CNN. However, in-order to avoid over fitting we kept the model comparatively shallow by using only 2 convolutions layers. Spiros V. Georgakopoulos et al. [2] identifies that CNN enhances the toxic content classification by emphasising on the structure of the words. The paper shows evidence that CNN outperforms well established methods for toxic text classification. The paper[5], describes for word representation, vector space having contextual meaning and explains how this outperforms traditional models. In our project we have used GloVe for representing text to maintain the contextual structure within each sentence. Pengfei Liu et al.[18] in his paper have used 16 different text classification technique in which LSTM with architecture similar to the one in Jozefowicz et al. paper[19] followed by softmax activation function has shown very promising result compared to others.

.

2 Method

2.1 Approach

In our project work, we first trained various traditional machine learning models using the widely used TF-IDF word vectorization technique to classify the sentences as either toxic or not. But these are not the best way to semantically classify texts as they do not preserve the context of each word while calculating the embedding. To overcome this drawback we used GloVe and supervised techniques for word to vector conversion to preserve the contextual meaning while converting each word into a vector. These vector were then feed into the deep learning architectures for classification. As the data was skewed in nature with majority of data tilting towards the non toxic section, so to minimize this under representation of toxic comments ,we used the random oversampling . The traditional approach and the deep architectures used are described below in detail.

2.1.1 Traditional approach

To begin with, in this approach we converted each word into a corresponding vector using TF-IDF. TF-IDF(Term Frequency Inverse Document Frequency) is to determine the weight and importance of each word in the corpus. More the number of times the word appears in the document, higher the TF-IDF value becomes. However the value is inversely proportional to the number of documents that contains the particular word, this makes sure that the more frequent appearing word are not given much weight[6].

After producing a vector for each sentence, these vectors are used as a input to traditional model unit. A logistic regression[12] uses sigmoid function and gives a output with confidence value for each sentence being toxic with values in range [0,1] where 1 indicates the sentence is likely more toxic. Random Forest classifier[13] is an ensemble of multiple deep decision trees trained on random parts of the same training sets. The results from these trees are averaged which gives values in range from [0,1] where 1 indicates more toxic sentence. A Naive bayes classifier[14] employ likelihood and naive bayes equation to produce posterior probability value for each class. The sentence belongs to class which has higher posterior probability. SVM[15] creates a hyperplane to segregate training data into toxic and non toxic classes. Depending on which side of the hyperplane the test sentence is present its class is determined.

The significant thing to consider here is that TF-IDF is naive approach which cannot distinguish between words of common root like 'cat' and 'cats' unlike the modern pre-processing approaches like stemming or lemmatization. It also cannot understand the relationship between the words. Also, the above traditional models are a very naive approach for classification.

2.1.2 Deep learning approach

Deep learning models i.e. ANN, CNN and LSTM trained on pertained and learned embeddings are shown to perform exceptionally well on NLP tasks[1][20][21][22]. So here we attempted to design a 4 layer MLP and CNN model and a 3 layer LSTM model to train it on already learned GloVe embedding and also on trained supervised embeddings for the problem domain. We further proceeded to tune the GLOVE embedding for our database for better resutls.

GloVe(Global Vectors) is unsupervised approach for obtaining vector representation for each word[4].Global corpus statistics are captured accurately by GloVe, hence earning this name. It uses two techniques that is Nearest neighbors and Linear substructures. For the nearest neighbour, the cosine distance or Euclidean distance is been used for measuring the similarity. At times, it finds out the uncommon relevant relationship. Linear substructures form a linear scales which tells the relatedness between the words. Vector difference in this helps to find juxtapositions words. We also produced a learned embedding during training which produces embedding for each word based on the sentence and the class of the sentence. The embedding produced using this method has relevant context to the the data we are training on.

The ANN[17] are a fully connected multi-layer neural network architecture in which each node performs the sum of all the weighted output from the previous layer nodes and then applies an activation function over the summation. This process is followed for each node in the network

beginning from the input layer and ending in the output layer. The ANN architecture that we implemented in our project is shown in the fig. 1.

The CNN[1] are special kind of artificial neural networks which contain convolutional layers. These convolutional layers contain a number of Filters which go over the input data scanning for a particular pattern. The next layer filters go over the output from the previous layers to extract even higher level features. In this way CNN extracts high level features from low level features which makes it extremely useful in classification of images and texts. The CNN architecture that we implemented in our project is shown in the fig. 1.

RNN(Recurrent Neural Network) models are powerful tools for NLP tasks as memory component is presented in each node[9]. LSTM is one variant of RNN which solves vanishing gradient problem,a major problem in most RNN model[8]. In this weight can not be updated as gradient tends to be zero. So the older node slowly ends up with no importance for the later outputs. It consists of 4 parts which are cell,input gate,output gate and forget gate. Gate are combination of sigmoid neural net layer and pointwise multiplication operation. It can remove or add information for the current node with help of this gate. The LSTM architecture that we implemented in our project is shown in the fig. 1.

For both the models we used ReLu activation function for all the nodes except the last layer where we used the sigmoid function. The final output go through the sigmoid activation function which tunes the output between 0 and 1. With higher the value of the output the larger the chance of it being a toxic comment.For LSTM, we have also add a dropout layer with a rate of 0.1 before using sigmoid function which makes sure that there is no over-dependence on particular nodes and overfitting is avoid. As we are doing binary classification, the cost function used here is binary cross-entropy. The weights for the model are learned through back propagation algorithm[10].

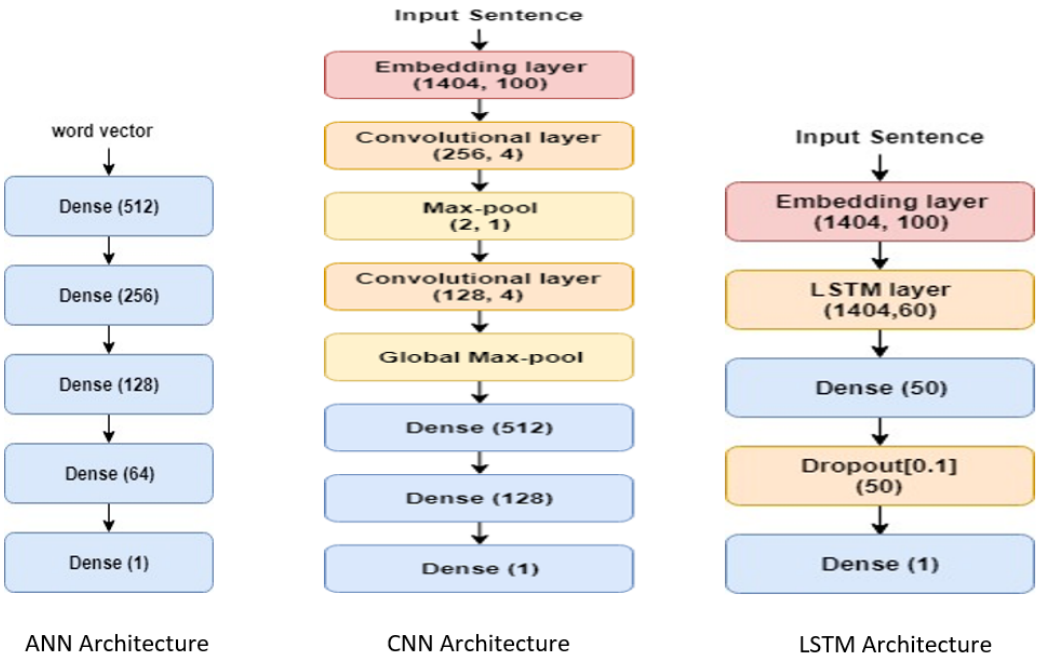

Figure 1: Deep Learning Architecture

2.2 Rationale

We initially used TF-IDF to convert the sentences into a vector representation which is necessary as machine learning algorithms generally works better with numbers. The TF-IDF is a great way for representing the importance of each word as they reduce the importance of the words that appear in most of the other sentences and give more importance to the words that are more specific to that sentence. This works decently but for efficient classification of texts we need to capture the semantic

pattern within the sentences.

To have a level of comparability, while designing out ANN and CNN models we restricted our self to only four layers excluding input and output layer. Inspired from [1] we increase the number of layers to increase the model complexity as we are training on more data at the same time not adding too many layers as then the model run the risk of overfitting[23]. After tuning the hyper parameters for the layers we came up with the architecture shown in fig 1 for ANN and CNN respectively.In LSTM, due to the lack of computation resources and to avoid overfitting, we constrained the number of layer to 3. It is structurally very similarly to ANN and CNN.

To compute the semantics meaning of each word we precalculated and learned embedding which depends on the context of the word rather than their frequency. In this way the representation is more rich and captures much more contextual information. For correctly classifying the sentences we need to identify the recurring semantic pattern within sentences. Deep neural networks like CNN are very apt at identifying patterns and have state of the art performance for various pattern recognition like image classification[11] and text classification[1]. Hence, here we are using a five layer convolutional neural network to learn the underlying semantic pattern within each sentences for correctly classifying the comments. CNN are also very robust against skewed data which makes the choice more relevant. We decided to have only two convolutional layers in our model to keep it relatively shallow and to avoid overfilling initially. Also, some of the comments containing bad words may not be toxic but actually constructive criticism or sarcasm, for CNN this pattern will be more apparent as compared to any other traditional machine learning approach which would not consider the context.

LSTM like any other RNN model are even better in classification of text due to the presence of memory components. With the Input gates, Output gates and Forget gate, it avoids the problem of long term dependencies. Thus improve the result even further than state of art RNN. Here we are using a 3 layer LSTM. Once the words are vectorized, it is pass through LSTM layer and 2 dense layer. The reason to keep this shallow is avoid overfitting. The dropout method is also used for the same reason. Hence with LSTM, the result should be much better than the traditional models.

3 Experiment

3.1 Data Description

The Data contains 159572 comments on which models were trained. The data set was taken from Kaggle and it was collected from Wikipedia comments which are labelled by human raters. A single tuple in the data has Identifier, Comment text and Binary representation for each type of toxicity. The toxicity of data is described as toxic, severe toxic, obscene, threat or insult. For our experiments we used Comment text which is textual in nature and type of toxicity as toxic. The data set is skewed as only 10 percent of the comments belongs to toxic class. We have divided data into 80 percent training set and 20 percent test set. The longest sentence in the data set has 1403 words, the average number of words in any sentence is 68 and the shortest sentence contains only an empty string.

3.2 Hypothesis

In our project, the research is focused in three main directions. Firstly, as we are dealing with subtle task of classifying toxic comments, it is highly significant to have more sensitivity in terms of correctly identifying comments which are toxic out of all the toxic ones. So we need to analyze how the deep learning models can be architectured to achieve this high sensitivity. Secondly, as mentioned above our data set in highly skewed towards non toxicity. This issue need to be sufficiently addressed when comparing different models' performance. Lastly, we need to obtain significant insights on the results obtained, for instance when the word is represented as vector using learned embeddings, the model should perform better compared to model representing word using TF-IDF. So, we need to analyze whether our models performance follows this expected norm.

3.3 Experimental Design

Firstly, we wanted to test if we can design deep learning architectures i.e ANN, CNN and LSTM that can perform better than the traditional models in classifying the minority class efficiently and also have a better overall performance. We designed three deep learning model i.e. Multilayer perceptron model, Convolutional neural network model and LSTM and trained them on the training set as that of the traditional models. These models were designed with the restrictions described in rationale above.

To answer whether using a learning vector representation for the data set would benefit as compared to using the traditional bag of words approach, we used two different pre-processing step for the same ANN architecture. In the first one we convert the data-set into a vector representation using the TF-IDF bag of words method and then feed this representation into the ANN for training. In the second one, we attached the embedding layer before the ANN architecture which converts the words into their vector representation and are learned along with the network. By comparing the performance of the two ANN in the presence of different pre-processing task, we can aptly judge if using learned embedding would benefit us or not. Further comparison can be drawn from these models if we used different sentences which contain bad words but are not toxic and visa versa for classification.

We used two different pre-processing steps with our CNN and LSTM models to answer whether using GLOVE will improve the overall performance or robustness for the minority class. For this we set up the two models with three different settings, one using the learned embedding second using the static GLOVE embedding and the third again using the GLOVE embedding but this time they are tuned as the model learns from the data-set. This will clarify whether tuning a unsupervised learned embedding would improve the performance on our data-set as claimed in many of the previous related studies.

To test whether the presence of more minority data point would make all the model more robust and accurate especially against the minority class classification. For this we over sample the minority class in our training data-set and use that to train all the models again for comparison. By observing the accuracy and F-score will give us a lot of information about the performance improvement with the balanced data-set.

In the above methods we divided 80% data into training set and remaining 20% to test set.

4 Results

All the traditional and deep learning model's hyper-parameters are tuned to give their best performance. Both the ANN and Convolutional Neural Network used was trained for 30 epoch whereas LSTM was trained for 10 epochs.

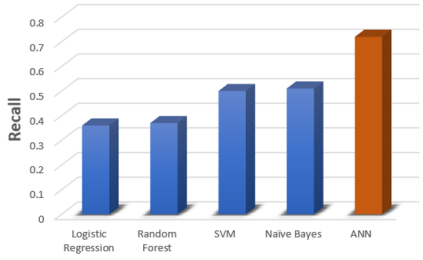

Figure 2: Traditional VS ANN Recall

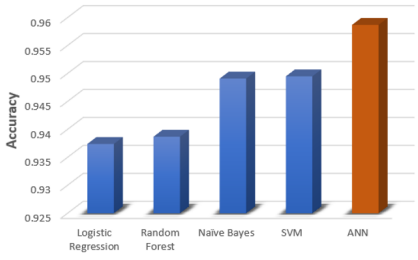

Figure 3: Traditional VS ANN Accuracy

Even with the data being highly skewed, traditional models struggled to perform good with achieving only 94.1% with naive bayes considering the majority class has more than 90% instances. Comparing this to our Multi-layer perceptron model using the same data pre-processing step on the sentences

significantly outperforms the traditional models as shown in figure 2 and 3. The robustness of the models are even more apparent when we compare the recall as shown in fig.2. Comparing the Recall clearly indicate the aptness of our four layer ANN to better learn to classify the minority class even with substantially less percentage of data points.

Models	Recall	Accuracy
CNN	0.7615	0.9512
LSTM	0.7721	0.9634
CNN with GloVe	0.7698	0.9487
LSTM with GloVe	0.7701	0.9602
CNN with GloVe Tuned	0.7743	0.9575
LSTM with GloVe Tuned	0.8823	0.9595

Table 1: CNN LSTM – GLOVE VS Learned Embedding

Models	Recall	Accuracy
ANN	0.7205	0.953
ANN with embeddings	0.7423	0.958

Table 2: Bag of words VS learned embeddings

As shown in table 2, even though our ANN seems to perform very well with the TF-IDF representation of the sentences in the data, its performance is further improved if we use the learned embedding to convert each word to a vector. These vectors preserves the context of the words by placing the vectors of similar meaning word closer together. In this way the ANN with embedding can look at the pattern in the entire sentences and decide whether it is toxic. And since the TF-IDF is a bag of word approach it doesn't preserve the context of the words,and this can lead to some interesting effects. To compare the correctness in classification of both the ANN models we constructed different sentences containing abhorrent words but are not toxic. Sure enough, the ANN using TF-IDF classified almost all of them as toxic, whereas the ANN using the learned embeddings showed low confidence in these sentences as being toxic most of the time. The ANN with TF-IDF even showed less confidence as compared to the other ANN for a sentence which contain threat but also have few good words. These comments and their confidence for both the models are shown in the table 3.

Coming to the CNN and LSTM architectures, LSTM seems to perform better than CNN in terms of both recall and accuracy. The thing to note here is when we used GLOVE embeddings we see an improvement in performance. There is further improvement in the performance when these GLOVE embedding are tuned for our data set while training as can be seen in table 1.

Sentence	ANN TF-IDF	ANN with Embedding
He had killed many people in the past and would continue to kill if we do not do something about it	0.921	0.288
In a time were rape and harassment are widespread, we need to stand up against the bullies.	0.918	0.464
The word **** is very inappropriate please refrain from using. It leaves a bad impression.	0.999	0.992
I will kill your family, nice good wonderful.	0.762	0.954

Table 3: Bag of words VS learned embeddings Classification Result

Models	Recall	Accuracy	F-Score
SVM	0.5	0.9495	0.65
ANN with embeddings	0.74	0.958	0.75
CNN with GloVe Tuned	0.77	0.9575	0.78
LSTM with GloVe Tuned	0.88	0.9595	0.79

Table 4: Traditional VS ANN VS CNN VS LSTM

The CNN and LSTM model performs the best out of all the deep learning models. As both of them do a great job in maintaining the semantic context of the sentences while classification. The CNN can extract high level semantic patterns from low level patterns using multiple convolution layers, hence better able to learn and classify the sentences. LSTM performs the best out of all the models as LSTM has memory component and this loop like structure can find out even more semantic patterns. As it can be seen in the table 4, deep learning models are very apt at classifying minority class efficiently even with less amount of data.

To address the issue with skewness of the data, we also trained our models after over-sampling the minority classes in the training data. As we had expected, the accuracy for every model went up, especially for the traditional models, the results of same can be shown in table 5. Unexpectedly, we see a great improvement in recall for traditional models especially SVM, which is almost as good as LSTM. But from precision it became clear that these models still lack the classification capability of the deep models. Traditional models are now identifying more toxic sentences but it is very poor at correctly classifying for toxic class.

Models	Recall	Accuracy	Precision	F-Score
Random Forest	0.77	0.9292	0.6	0.68
SVM	0.86	0.9356	0.62	0.72
CNN with GloVe Tuned	0.76	0.9579	0.79	0.78
LSTM with Glove Tuned	0.86	0.9462	0.67	0.75

Table 5: Over Sampling Minority Classes

5 Conclusions

After research, analysis and creating models using our data, we found that the Multi-layer perceptron models outperform traditional models in accuracy and robustness even with the highly skewed data. We observed that ANN with learned embedding performs well in understanding the context of the sentence and correctly classifies constructive criticism as non toxic.

We also found that our CNN and LSTM model outperforms all other deep learning models as they correctly classify data semantically and they perform best when the words are converted to vector using GLOVE embeddings which are tuned for our data set. This becomes very evident from the results for LSTM which achieved 86% recall with a great F-score when trained using GLOVE embeddig tuned for our data-set. These deep learning models are more robust against skewed data and hence, oversampling minority classes along with deep learning architecture performs the best in terms of accuracy, precision and recall.

Future Scope : The current deep neural network model performs very well, but we believe it can be further enhanced by increasing the depth of our models to achieve a better recall and F-Score.

6 References

[1] Kim, Yoon. "Convolutional neural networks for sentence classification." *arXiv preprint arXiv:1408.5882 (2014)*

[2] Georgakopoulos, S.V. & Tasoulis, S.K & Vrahatis A.G & Plagianakos V.P "Convolutional Neural Networks for Toxic Comment Classification." *In Proceedings of the 10th Hellenic Conference on Artificial Intelligence (p. 35). ACM. 2018*

[3] Ibrahim, Mai, Marwan Torki, and Nagwa El-Makky. "Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning." *17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2018.*

[4] van Aken, Betty, et al. "Challenges for Toxic Comment Classification: An In-Depth Error Analysis." *arXiv preprint arXiv:1809.07572 (2018).*

[5] Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.*

[6] Word Embedding Wikipedia

[7] Ramos, Juan. "Using tf-idf to determine word relevance in document queries." *Proceedings of the first instructional conference on machine learning. Vol. 242. 2003.*

[8] Long Short-term Memory by Sepp Hochreiter(1997)

[9] Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078 (2014).*

[10] Hecht-Nielsen, Robert. "Theory of the backpropagation neural network." *Neural networks for perception. Academic Press, 1992. 65-93.*

[11] Krizhevsky, A. & Sutskever, I. & Hinton, G.E. "ImageNet Classification with Deep Convolutional" Neural Networks." *Advances in neural information processing systems. 2012.*

[12] Ifrim, Georgiana, Gökhan Bakir, and Gerhard Weikum. "Fast logistic regression for text categorization with variable-length ngrams." *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008*

[13] S. Bharathidason, C. Jothi Venkataeswaran. "Improving Classification Accuracy based on Random Forest Model with Uncorrelated High Performing Trees". *International Journal of Computer Applications (0975 – 8887),Volume 101– No.13, September 2014*

[14] Kim, Sang-Bum, et al. "Some effective techniques for naive bayes text classification." *IEEE transactions on knowledge and data engineering 18.11 (2006): 1457-1466*

[15] Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." *European conference on machine learning. Springer, Berlin, Heidelberg, 1998*

[16] Das, Bijoyan, and Sarit Chakraborty. "An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation." *arXiv preprint arXiv:1806.06407 (2018).*

[17] Xin Yao. "Evolving artificial neural networks." *Proceedings of the IEEE (Volume: 87 , Issue: 9 , Sep 1999)*

[18] Liu, Pengfei, Xipeng Qiu, and Xuanjing Huang. "Adversarial multi-task learning for text classification." *arXiv preprint arXiv:1704.05742 (2017).*

[19] Jozefowicz, Rafal, Wojciech Zaremba, and Ilya Sutskever. "An empirical exploration of recurrent network architectures." *International Conference on Machine Learning. 2015.*

[20] Lee, Ji Young, and Franck Dernoncourt."Sequential short-text classification with recurrent and convolutional neural networks." *arXiv preprint arXiv:1603.03827 (2016)*

[21] Yin, Wenpeng, et al. "Comparative study of cnn and rnn for natural language processing." *arXiv preprint arXiv:1702.01923 (2017)*

[22] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781 (2013)*

[23] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, Dhruv Batra. "Reducing Overfitting in Deep Networks by Decorrelating Representations." *arXiv preprint arXiv:1511.06068 (2016).*

GITHUB URL - https://github.com/akanksha0625/TOXIC-CONTENT-CLASSIFIER