CONTENT MODERATION USING DEEP LEARNING

Minor Project II

Submitted by

VRINDA GOYAL (9917103238)
ANJALI KUMARI (9917103248)
TUSHIT GARG (9917103259)

Under the supervision of

Ms. Anuradha Gupta

Department of CSE/IT
Jaypee Institute of Information Technology University, Noida

MARCH 2020

ABSTRACT

Social Networks provide a platform to its users to express their view, thought process and opinion. Millions of users are connecting with Facebook, Twitter, Linkedin, Reddit and, many other social network platforms. With the ease in communication, it also brings about important challenges. Since sensitive content on a platform negatively affects user experience, communities have terms of usage or community norms in place, which when violated by a user, leads to moderation action on that used by the platform. Unfortunately, the scale at which these platforms operate makes manual content moderation nearly impossible, leading to the need for automated or semi-automated content moderation systems. In our project, we would use deep learning models to correctly predict the “hate” class. It is quite unclear which model is most suitable for a certain platform since there have been few benchmarking efforts for moderated content. To that end, we compare existing approaches used for automatic moderation of multimodal content on three online platforms: Twitter, Reddit, Quora. In practical scenarios, labeling large scale data for training new models for a different domain or platform is a cumbersome task. Therefore we combine our existing pre-trained model with a minimal number of labeled examples from a different platform to create a co-trained model for the new platform. We perform a cross-platform analysis using different models to identify which model is better. Using co-trained models would be a cost-effective solution.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

1.	WORD2VEC	Word to Vector
2.	LSTM	Long-short term memory
3.	GRU	Gated-Recurrent Units
4.	RNN	Recurrent Neural Networks
5.	CNN	Convolutional Neural Networks
6.	BoWV	Bag of Word Vectors
7.	GBDT	Gradient Boosted Decision Trees
8.	SVM	Support Vector Machine
9.	LDA	Linear Discriminated Analysis
10.	PCA	Principal Component Analysis
11.	DTM	Document Term Matrix

INTRODUCTION

Social media sites are platforms that showcase user-generated content to engage participants. These participants are provided an abundance of reach, freedom to express their opinions and receive feedback at marginal cost. This online content covers each and every minute detail of a user on a daily basis. Despite the huge benefits of social media, it also brings along unique challenges. Often, users encounter issues like cyberbullying, online threats, abuse, harassment and hate speech. There has been an enormous increase in objectionable content on different social media platforms. This daily onslaught of disturbing posts can lead to conflicts on the web. Such content needs to be pulled down from websites. This is where content moderation comes into the picture. It is an important and relevant problem as many platforms are struggling to solve it.

Problem Statement

- There has been an enormous increase in objectionable content on different social media platforms. This daily onslaught of disturbing posts can lead to conflicts on the web. Such content needs to be pulled down from the websites. This is where Content Moderation comes into the picture.
- The scale at which these platforms operate makes manual content moderation nearly impossible, needing to the need for automated or semi-automated content moderation systems.
- Instead of using different models for different platforms, a cross- platform analysis would be an effective solution.

Why it is important?

Sensitive content is a problem for almost every platform. Can a single model be used to tackle the problem on different platforms like Reddit or Twitter. If not, then it is essential to investigate the reasons for variable performance and propose changes that can be made to the model to improve performance on other platforms. Furthermore, comparisons between existing automated content moderation techniques can help us understand the limitations of existing methods and identify gaps.

BACKGROUND STUDY

RESEARCH PAPERS:

PAPER 1

Title of the Paper: Deep Learning for Hate Speech Detection in Tweets

Author: Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma

Summary: This paper focuses on the problem of classifying a tweet as racist, sexist or neither. The task is quite challenging due to the inherent complexity of the natural language constructs – different forms of hatred, different kinds of targets, different ways of representing the same meaning. They have experimented with multiple classifiers such as Logistic Regression, Random Forest, SVM, Gradient Boosted Decision Trees(GBDT) and Deep Neural Networks, three deep learning architectures: FastText, Convolutional Neural Networks(CNN) and Long Short Term Memory(LSTM) networks with feature spaces comprising of character n-grams, TF-IDF vectors, and Bag of Word vectors(BoWV). They have initialized the word embeddings with random and GloVe embeddings, performed 10- fold cross-validation and calculated weighted macro precision, recall, and F1 scores. Finally, they concluded that among baseline methods TF-IDF method is better than char n-grams, CNN performed better than LSTM which was better than FastText, initialization with random embeddings was slightly better than with GloVe embeddings and the best method was LSTM+ Random Embeddings + GBDT. Also, embeddings learned from deep neural network models when combined with gradient boosted decision trees led to the best accuracy values.

PAPER 2

Title of the Paper: Hateful symbols or Hateful People? Predictive Features for hate speech detection on Twitter

Author: Zeerak Waseem and Dirk Hovy

Summary: Hate speech in the form of racist and sexist remarks are a common occurrence on social media, this paper provides a list of criteria founded in critical race theory and uses them to annotate a publicly available corpus of more than 16k tweets. They bootstrapped the corpus collection by performing an initial manual search of common slurs and terms used about religious, sexual, gender and ethnic minorities. Gender details were extracted by looking up names in the user's profile text, the name, or the username provided and compared to known male and female names as well as other indicators of gender and hence found out gender distributions to be heavily skewed towards men. Further, they normalized the data by removing stop words and constructed the ten most frequently occurring words. They found out that using location as a feature

negatively impacted the F1 score and thus used time zone as a feature for classification. To evaluate the influence of different features on prediction in a classification task, logistic regression classifier and 10- fold cross-validation was used. To pick the most suitable features, a grid search was performed over all possible feature set combinations and it was found that character n-grams outperformed word n-grams. Uni, bi, tri and four- grams for each tweet and the user description were collected and it was concluded that the F1 score decreases by the use of all features namely Gender + location + length.

PAPER 3

Title of the Paper: Convolutional Neural Networks for Toxic Comment Classification

Author: Spiros V. Georgakopoulos, Sotiris K. Tasoulis, Aristidis G. Vrahatis, and Vassilis P. Plagianakos

Summary: This paper employs the use of CNN to discover toxic comments in a large pool of documents and compare CNN's against the traditional bag-of-words approach. This paper details the methodology for using CNN in text classification beginning from the creation of the vocabulary, encoding the documents and transforming them into matrices and later passing them through the embedding layer. Also, a BoW model is described as an alternative way of extracting features from text wherein we construct the Document-Term-Matrix (DTM), create the vocabulary and validate the DTM using Principal Component Analysis (PCA) and t-SNE. Word embeddings and CNN are compared against the BoW approach for which four well-established text classification methods namely SVM, Naive Bayes (NB), k-Nearest Neighbor (kNN) and Linear Discriminated Analysis (LDA) are applied on the designed DTMs. Finally, a statistical analysis is performed on the outcomes of the binary classification task and a confusion matrix is created. It was found that CNN models outperformed SVM, kNN, NB, and LDA achieving accuracy almost over 90%, the lowest false discovery ratio and lowest variance with the best performance w.r.t to precision and recall. It was concluded that CNN outperforms traditional text mining approaches for toxic comment classification presenting great potential for further development in toxic comment identification.

PAPER 4

Title of the Paper: Aggression Identification using Deep Learning and Data Augmentation

Author: Julian Risch and Ralf Krestel

Summary: This paper proposed to augment the provided dataset to increase the number of labeled comments from 15000 to 60000. Their approach is based on a recurrent neural network, a bi-directional gated recurrent unit (GRU) layer with max pooling and average pooling. Machine translation has been employed to augment the dataset. Machine translating a user comment into a foreign language and then translating it

back to the initial language preserves its meaning but results in different wording which served their approach that if the translation did not change the wording, it could not augment the dataset. However, because the wording is different the translated comment adds to our dataset. Only if the meaning is preserved, we can assume that the label (non-aggressive, covertly aggressive, overtly aggressive) of the initial comment also holds for the translated comment. Required pre-processing was done by splitting the words through a dynamic programming approach, pre-trained, fixed FastText embeddings were used. TF-IDF was applied to the character and word n-grams and along with a few handpicked features, an ensemble was created. For each comment, the length, the relative number of uppercase characters, non-alpha characters and exclamation marks performed equally well on the English dataset.

DETAILED DESIGN

The methodology for analyzing public opinion incorporates (1) data collection, (2) pre-processing, (3) feature extraction (4) visualizes data.

Data is downloaded from Twitter using Twitter streaming API, from Reddit using Reddit API and Quora from a dataset available on Kaggle and stored in **MongoDB** database which helps in eliminating duplicate entries and helps in fast queries. The data is then pre-processed and cleaned into a trimmed data set in JSON format. Feature extraction (word2vec, word embeddings, FastText, GloVe) is done for text classification. These features serve as input to machine learning classifiers.

Steps Involved:

3.1. Data collection:

The first step was to collect real-time data for the study. We chose Twitter, Reddit, and Quora as platforms to collect the data related to Hate Speech. For this, we have used Twitter and Reddit streaming APIs and Quora's dataset which provides the related feed in a machine-readable JSON format.

3.2 Data PreProcessing

After getting the dataset that contains data related to hate speech, the next step was to clean the data to provide the input for the text classification model. Accuracy of feature extraction also greatly depends on the quality of text data. The following are the steps performed for data cleaning.

- ***Normalization***

This refers to the conversion of any non-text information into the textual equivalent. For this, we have used a library called normalize. This library is based on nltk package, so it expects nltk word tokens.

- ***Removal of Punctuation marks and symbols***

  A regular expression is used to eliminate the unnecessary punctuation marks(,;!'".?/*-....etc) and symbols like emojis, emoticons. removed. URLs, extra line feeds.

- **Tokenization and Removal of Stop Words**
  - *Tokenize*:

    This breaks up the strings into a list of words or pieces based on a specified pattern using Regular Expressions.
  - *Stop Words*:

Stop words are generally the most common words(such as "the", "a", "an", "in") in a language. These words are of no use because they don't help us to find the context or the true meaning of a sentence. We would not

want these words to take up space in our database, or taking up the valuable processing time. These words were removed from the previously cleaned tweet text using a famous NLP library **Spacy.**

- **Stemming and Lemmatization**

  Stemming and Lemmatization is Text Normalization techniques in the field of NLP that are used to prepare text, words, and documents for further processing. Lemmatization in linguistics is the process of grouping together the inflected forms of a word so they can be analyzed as a single item, identified by the word's lemma, or dictionary form.

  Lemmatization of the words is done using the spacy library.

3.3 Feature Extraction:

3.3.1 WORD EMBEDDING

It's a representation of text where words that have the same meaning have a similar representation. In other words, it represents words in a coordinate system where related words, based on a corpus of relationships, are placed closer together. In the deep learning frameworks such as TensorFlow, Keras, this part is usually handled by an **embedding layer** which stores a lookup table to map the words represented by numeric indexes to their dense vector representations.

3.3.2 WORD2VEC

Gensim implementation of the word2vec model is used for the training of the dataset. The first step is to prepare the text corpus for learning the embedding by creating word tokens, removing punctuation, removing stop words, etc. This is done by the Tokenizer function of Keras. The word vector thus generated is passed to the gensim.models. Word2Vec function which trains the word embedding on the dataset. The model then uses this pre-trained word2vec embedding for the classification of tweets.

3.3.3 GLoVE

GloVe stands for global vectors for word representation. It is an unsupervised learning algorithm for generating word embeddings by aggregating a global word-word co-occurrence matrix from a corpus. The resulting embeddings show interesting linear substructures of the word in vector space.

3.3.4 FastText

FastText is an extension to Word2Vec proposed by Facebook in 2016. Instead of feeding individual words into the Neural Network, FastText breaks words into several n-grams (sub-words). For instance, the tri-grams

for the word apple is app, ppl, and ple (ignoring the starting and ending of boundaries of words). The word embedding vector for apple will be the sum of all these n-grams. After training the Neural Network, we will have word embeddings for all the n-grams given the training dataset. Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words.

IMPLEMENTATION

4.1 Classification Using Deep learning

Deep learning is a subset of machine learning in artificial intelligence (AI) that has networks capable of learning unsupervised from data that is unstructured or unlabelled. Also known as deep neural learning or deep neural network. Deep learning text classification model architectures generally consist of the following components connected in sequence:

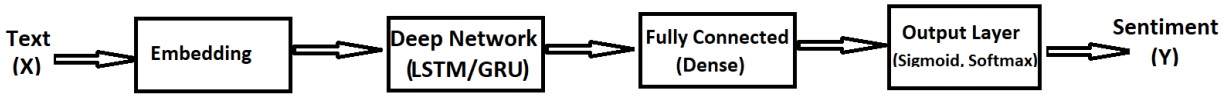

Fig 1.1 Deep learning architecture for text classification

Deep Learning architecture consists of layers with each layer's output working as an input to the next layer. As seen in the above diagram, the text is passed to the first layer which is the embedding layer, which then passes to the next layer and so on. To implement this architecture, we have used **Keras,** an open-source neural network library written in Python with a background working on TensorFlow.

4.1.1 Implementation

For text classification using deep learning architecture we have used the following layers:

Layer1: Embedding Layer

This layer makes use of pre-trained word embeddings (in our case, trained on our dataset). The input to this layer consists of the embedding matrix which was made using the trained word embedding on our dataset.

Layer2:

Multilayer Perceptron:

Sequential() imported from Keras acts as a hidden layer between the input and the output layer.

Gated recurrent units:

Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks. The GRU is like LSTM with forget gate, but has fewer parameters than LSTM, as it lacks an output gate.

Parameters used :

- units=16
- dropout=0.2
- recurrent_dropout=0.2

Recurrent neural networks:

Recurrent Neural Network(RNN) is a type of neural network where the output from the previous step are fed as input to the current step.RNN is used to remember some information about a sequence.

Parameters used:

Long-short term memory:

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. LSTM unit is composed of a cell, an input gate, an output gate, and a forget gate. The cell remembers values over arbitrary time intervals and the three *gates* regulate the flow of information into and out of the cell.

Concurrent neural network:

The first layers embed words into low-dimensional vectors. The next layer performs convolutions over the embedded word vectors using multiple filter sizes. For example, sliding over 3, 4 or 5 words at a time. Next, we max-pool the result of the convolutional layer into a long feature vector, add dropout regularization, and classify the result using a softmax layer.

Layer3: Output Layer

For binary classification, the activation function used is **'sigmoid'** and for multiclass classification **'softmax'** function is used. The number of output nodes depends on whether binary or multiclass classification is done. After the layers are made, the compilation of the model is done. Compile choices used in our model:

- optimizer='adam'
- loss='binary_crossentropy'
- metrics='accuracy'

After compilation, the model was fit on a training data of 50% and validation and testing data of 25% each.

- Batch_size = 32
- Epochs = 10

EXPERIMENTAL RESULTS AND ANALYSIS

5.1 Results and analysis of Machine Learning model

Evaluation Model

Precision - Precision is the ratio of correctly predicted positive observations of the total predicted positive observations. Precision = TP/TP+FP. .

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to all observations in actual class - yes. Recall = TP/TP+FN .

F1 Score- F1 score is the weighted average of Precision and Recall. F1 Score = 2*(Recall * Precision) / (Recall + Precision) .

MODEL	FEATURE INPUT	ACCURACY	PRECISION	RECALL	F1 SCORE
CNN	Word2Vec	79.2%	85.2%	29.2%	43.5%
	Glove	83.3%	82.1%	50.2%	62.3%
	FastText	80.4%	85.3%	34.6%	49.2%
MLP	Word2Vec	77.8%	85.1%	23.1%	36.3%
	Glove	80.9%	73.3%	47.8%	57.9%
	FastText	77.9%	72.2%	31.5%	43.9%
LSTM	Word2Vec	79.6%	91.0%	28.4%	43.3%
	Glove	84.0%	77.7%	58.4%	66.7%
	FastText	80.5%	76.2%	41.9%	54.0%
GRU	Word2Vec	80.3%	84.1%	34.7%	49.1%
	Glove	84.6%	83.5%	54.6%	66.0%
	FastText	81.1%	90.8%	34.6%	50.1%
RNN	Word2Vec	77.1%	89.0%	16.0%	35.8%
	Glove	81.5%	77.7%	45.8%	57.6%
	FastText	79.2%	82.3%	29.5%	48.0%

Fig 1.2 Accuracy and Loss graphs for training and validation (Model - LSTM, Feature Input-Glove)

CONCLUSION

In this project, we have evaluated several deep learning approaches and identified those most suitable for detecting sensitive content. We have shown it is possible to analyze a large body of social media data using deep learning in a reliable and replicable way by employing a methodology to collect, train and classify data. We also compared the accuracy of different deep learning models and found out that LSTM performs best with GloVe word embeddings on our dataset.

PROPOSED WORK PLAN FOR FUTURE

We would further apply all the deep learning models on the rest of the platforms and would perform a multi-platform comparison and analyze which model best suits the platform. Also, we would perform a cross-platform content-moderation using a co-trained model enriched with a small amount of labeled data from a different platform, to moderate content on that platform. A multimodal approach is used for image and text datasets. Finally, we would deploy a portal as a proof-of-concept for different methods.

REFERENCES

Research Papers :

[1] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In SRW@HLT-NAACL, 2016.

[2] Spiros V. Georgakopoulos, Sotiris K.Tasoulis, Aristidis G. Vrahatis, and Vassilis P. Plagianakos. Convolutional neural networks for toxic comment classification. In SETN, 2018.

[3] Hasso Plattner. Aggression identification using deep learning and data augmentation. 2018.

[4] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. In WWW, 2017.

E-Books :

[1] Natural Language Processing by Python by Steven Bird, Ewan Klein, and Edward Loper

[2] Deep Learning by Ian Goodfellow, Yoshua Bengio, Aaron Courville

Website Links :

[1] https://www.kaggle.com/c/quora-insincere-questions-classification/data.

[2] https://www.reddit.com/r/ListOfSubreddits/wiki/listofsubreddits

[3] The positives of social media: Spread of information. http://lifeasoflate.com/2013/11/the-positives-of-social-media-spread-of-information.html, 2013.