Sentiment Analysis using Convolutional Neural Network

1st Palak Bhatia
Student Id: 1116552
Lakehead University

2nd Pinak Divecha
Student Id: 1101608
Lakehead University

3rd Shiv Hansoti
Student Id: 1105615
Lakehead University

4th Zalak Manani
Student Id: 1105524
Lakehead University

***Abstract*—Sentiment analysis is acknowledged as one of the most significant sub-areas of Natural Language Processing (NLP) research where comprehension of implied or explicit emotions conveyed online is beneficial to consumers, company owners, and other stakeholders. It provides a simple consumer view, which may be something that carries a contextual conclusion, such as an internet analysis, reviews on blog entries, film ranking, and so on. Inspired by deep learning results, we are interested in using deep learning models to tackle the challenge of sentiment analysis. This paper targets to implement a scalable and robust Convolutional Neural Network-based solution for the problem of movie based sentiment analysis. The dataset used is IMDB movie reviews from the NLTK library that consists of 50,000 reviews.**

I. Introduction

Sentiment Analysis is the algorithmic treatment of opinion, sentiment and subjectivity and can be classified as a shared field of both Natural Language Processing (NLP) and Machine Learning (ML) within the broadly defined Artificial Intelligence (AI) field. At least part of a recent rise in activity in this area can be attributed to the interest in dealing directly with opinions as a tangible commodity in areas including, but not limited to, marketing, politics, decision support systems and finance. Countless studies have shown that online consumer reviews have a huge impact on the behavior of off-line purchases. Consumers are also willing to pay more for five-star rating goods from 20% to 99% than a four-star rated product.

The advent of deep learning approaches has given rise to several new ways of analysing sentiments. The use of broad unlabeled textual data can be used for studying the meanings of words and the nature of the creation of sentences. Word2Vec has attempted this by studying word embedding from unlabeled samples of document. It learns both by predicting the word and surrounding terms (SKIP-GRAM) from the given word. Such terms embedding are used to construct dictionaries, and traditional methods such as Term Frequency-Inverse Document Frequency (TF-IDF) [1] etc serve as dimensionality reducers. Further methods are seen capturing representations of the sentence points, such as the Recurrent Neural Tensor Network (RNTN) [2].

Neural Convolution Network primarily used for image-related tasks has also been shown to be effective in text classification. The primary challenge is the natural language's variable length. Some of it is addressed by context windows of fixed size but it fails to capture dependencies that span further than the context window. Recurrent Neural Network (RNN) may take variable duration of text series, but learning is extremely tricky. So new types of RNN such as Long Short Term Memory (LSTM) [3] and Gated Recurrent Unit (GRU) were invented. LSTM was proposed by Hochreiter et al. in 1997 and is producing news for many NLP tasks, such as sentiment analysis, translation and generation of sequences. The comments or review which is taken as an input is referred as a document. This document can't be directly used for sentiment classification hence pre-processing is need to be done after which any we can use convolutional neural network.

II. Literature Review

Sentiment analysis of the content created by online users is essential for many tasks relating to social media analytics. In the context of social media, there are several challenges. First, there are enormous quantities of data. Second, social networking posts are casual and short by definition. Analysis of opinion is, therefore, a very difficult task. Researchers from the natural language processing and information retrieval have developed many different approaches to solving sentiment analysis, most of which use Bag of Words (BoW) representations in this area. Because of the recent achievement of deep learning, more and more researchers are using deep learning algorithms to solve the demanding task of sentiment analysis.

The Recursive Neural Network (RNN) is a popular, deep learning system given by Socher [4]. Socher used completely labeled parse trees to reflect the rottentomatoes.com website reviews of the films. Recursive neural models simulate parent node vectors using the node vectors of the two children to measure the parent vectors. At last, the parse tree's root node can represent the sentence. This is the Recursive Neural Network's simplest member of the family suggested by Scocher. He suggested an improved Recursive Neural Network called Recursive Neural Tensor Network (RNTN) in 2013 [5]. The main concept is to use the same compositional function based on a tensor for all nodes. In other words, by adding one parameter to the models, they take into account the distance of the word in a sentence. Nonetheless, both RNN and RNTN have a downside. Both need completely labeled parse trees to train the neural network, so having of word and group of words will be a lot of heavy jobs. The Convolutional Neural Network, on the contrary, only requires the labels of the entire sentences, rather than the completely labeled parse tree. At the

same time, CNN has far less links and criteria, and is easier to train.

The Convolutional Neural Network (CNN) [6] has proved to be very successful in solving computer vision-related tasks. CNN includes layers of convolution and layers of pooling. In a general case, it is accompanied by the pooling layers integrating the outputs of the clusters of neurons [7]. However, when faced with more complicated problems, CNN's range and complexity would continue to expand and will be constrained by computing resources. Recently, training a broad deeply convolutional neural network has become possible thanks to the increasingly powerful GPU computing. In 2012, Dan Ciresan et al. significantly improved upon the best performance in the literature for multiple image databases, including the MNIST database, the NORB database, the HWDB1.0 dataset (Chinese characters), the CIFAR10 dataset (dataset of 60000 32x32 labeled RGB images) [8], and the ImageNet dataset [9]. CNN models have subsequently been shown to be useful for the sentiment analysis problem. N. Kalchbrenner described a Dynamic Convolutional Neural Network (DCNN) [10] for the semantic modeling of sentences. The network used Dynamic k-Max Pooling, a global pooling operation over linear sequences.

III. Proposed Model

In our proposed model, we are using publicly available movie reviews data-set. We divide the whole data-set into test and train set. We pre-process the train data by doing tokenization, removing stop words, lemmitization. After doing pre-processing, we do vectorization after which it becomes the input for our Convolution Neural Network along with the target attribute that is 'sentiment value'. Once the model is trained, we test our model using the testing data. This whole process from loading the data to training the model and than testing the model is shown in the block diagram which we have followed for implementing our project as in Fig 1.

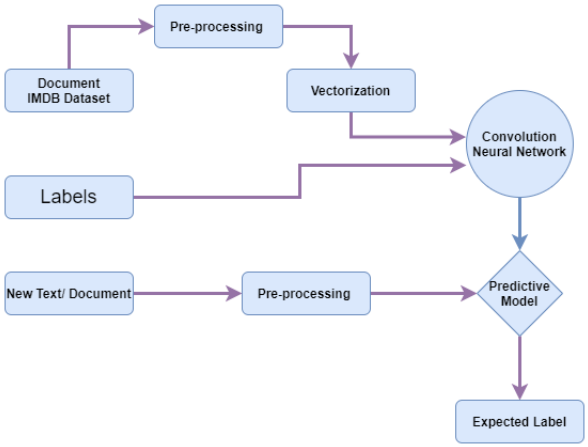

Fig. 1. Block Diagram

A. Dataset

In this project we have used publicly available movie reviews data sets that is IMDb reviews. IMDb (Internet Movie Database) is an online database of information related to films, television programs, home videos, video games, and streaming content online – including cast, production crew and personal biographies, plot summaries, trivia, fan and critical reviews, and ratings. Originally a fan-operated website, the database is owned and operated by IMDb.com, Inc., a subsidiary of Amazon. It contains total of 50000 reviews, out of which we would be splitting the dataset into 70:30 ratio that is 35000 for training and remaining 15000 for testing. A label in this case is a binary flag indicating if the review is positive (1) or negative (0). There are two attributes in the used IMDb dataset:

- Review: It is the group of words that reviews the particular movie.
- Sentiment: It states the sentiment as 'positive' or 'negative' for any given review.

B. Libraries

We have used the torchtext library for importing the data and performing intial pre-processing of the dataset. We have used spaCy for tokenization. SpaCy is a free, open-source library for advanced Natural Language Processing (NLP) in Python. We have used the torchtext library to perform the split over dataset into 70:30 with a random state set to 2003. Finally, we have made use of pytorch library to build the Convolution neural network and train our model.

C. Trainable Hyper Parameters

Various hyper-parameters have been used to tune in the model which are as discussed below:

- Kernel:- In machine learning, a "kernel" is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. The model uses a kernel size of 3 for the first convolutional layer and 5 for the second convolutional layer to achieve notable accuracy.
- Batch Size:- A smaller mini-batch size (not too small) usually leads not only to a smaller number of iterations of a training algorithm, than a large batch size, but also to a higher accuracy overall, i.e, a neural network that performs better, in the same amount of training time, or less. Therefore, the batch size used in the model is 64.
- Number of Epochs:- The number of epochs is the number of complete passes through the training dataset. It is set to 12 for the model since the accuracy was not notably affected even if we keep on increasing the epochs.
- Stride Rate:- In CNNs, a stride represents the number of shifts of pixels in the input matrix. Herein, no stride rate is used by the model.
- Learning rate:- The amount that the weights are updated during training is referred to as the step size or the "learning rate." Specifically, the learning rate is a configurable hyper-parameter used in the training of neural networks

that has a small positive value, often in the range between 0.0 and 1.0 . For this model, the learning rate is set to 0.01.

- Optimizer:- Optimizers are algorithms or methods used to change the attributes of your neural network such as weights and learning rate in order to reduce the losses. Adam optimizer has been used in the model.

Table-I shows the brief of the hyper parameters with their values which we have used in our model:

TABLE I
HYPER-PARAMETERS FOR PROPOSED MODEL

HYPER PARAMETERS	
Kernel size	3 -Layer 1 5 - Layer 2
Epochs	12
Batch size	64
Learning rate	0.01
Optimizer	Adam

D. Method: Convolution Neural Network

In this section, we implemented to solve the proposed Movie Review Sentiment Analysis tasks.
A recent breakthrough in the field of natural language processing is called word embedding. This is a technique where words are encoded as real-valued vectors in a high-dimensional space, where the similarity between words in terms of meaning translates to closeness in the vector space. Discrete words are mapped to vectors of continuous numbers. This is useful when working with natural language problems with neural networks and deep learning models. Pytorch provides a convenient way to convert positive integer representations of words into a word embedding by an Embedding layer.

Our model consists of a Embedding layer and three convolution layers followed by max pooling layers. After that, we have a dropout layer (with rate = 0.5) followed by a flattening layer and finally a dense layer which uses softmax activation function. The first convolution layer uses 100 filters having kernel size of 3x3, stride rate of 1. The second convolution layer has 64 filters with kernel size of 4x4. The third convolution layer has 64 filters with kernel size of 5x5. The model summary is shown below in Fig 2.

```
CNN(
  (embedding): Embedding(25002, 100)
  (conv_0): Conv2d(1, 100, kernel_size=(3, 100), stride=(1, 1))
  (conv_1): Conv2d(1, 100, kernel_size=(4, 100), stride=(1, 1))
  (conv_2): Conv2d(1, 100, kernel_size=(5, 100), stride=(1, 1))
  (fc): Linear(in_features=300, out_features=1, bias=True)
  (dropout): Dropout(p=0.5, inplace=False)
)
```

Fig. 2. Model Diagram

E. Inference Time

In machine learning and NLP domain inference time is the time taken by the trained model to infer/predict the testing array or samples. We have used python library to calculate the inference time that is the time taken to train the model. In order to do that we have the 'timeit' library. On an average, model takes around twelve minutes to build the model.

F. Model Storing

After creating the CNN model, we have stored the model into the drive. In order to save the model, we have first connected the drive with the colab. Once the drive was mounted, we have stored the model with .pt extension. We have also tested the saved model by loading the model again and testing it using the test instances.

IV. EXPERIMENTAL ANALYSIS

While training the model, we altered various training parameters such as kernel size, padding, stride rate, batch size, learning rate, number of epochs etc to achieve higher accuracy. Along with the training parameters, we also changed the optimizers and activation functions to predict the model with highest testing accuracy. We also tried to compare the accuracy by adding and removing various layers. Also, we have changed various pre-processing steps to achieve higher accuracy. Some of these comparisons have been analyzed below.

And so after designing and training our CNN model for sentiment analysis, firstly we have recorded various outputs. We have herein used Adam, Rprop and Adadelta optimizers. For each of them, we have changed the number of epochs that gave different percentage of accuracy and loss which can all be noted in the table-II and it can be clearly seen that Adam is the perfect fit for our model.

TABLE II
EXPERIMENTAL ACCURACIES

ACCURACY COMPARISON			
Optimizer	Epochs	Accuracy	Loss
Adam	5	85.68%	0.337
	10	85.05%	0.378
Rprop	5	84.77%	1.037
	10	84.40%	0.650
Adadelta	5	84.20%	0.401
	10	84.93%	0.431

Secondly, we have tried to note the outputs by variating the batch size for Adam optimizer and keeping epoch value as five. Table-III clearly describes the values of accuracy and loss for the same.

TABLE III
VARIATIONS FOR ADAM OPTIMIZER

ADAM OPTIMIZER		
Batch size	Accuracy	Loss
34	85.34%	0.340
64	85.40%	0.340
128	85.93%	0.355

Table-IV shows the output of the trained model to predict the sentiment of user comment on movies. First column is the example of the sentences used for testing the trained model. Second column describes the polarity of sentence that they

are either positive, negative or neutral. And the final column describes the Human prediction of the sentence that either the comment of the movies are positive or negative or neutral.

TABLE IV
SENTENCE POLARITY ILLUSTRATION

SENTENCE POLARITY		
Sentence	Polarity	Human Prediction
This film is boring	0.04	Negative
please don't watch that movie it is boring	0.29	Negative
That movie is worst	0.04	Negative
that movie is worse than previous one	0.03	Negative
This movie is not that bad	0.45	Neutral
This film is great	0.94	Positive
This movie is awesome	0.83	Positive
I love this movie	0.70	Positive

We have plotted the graph of Epoch vs Training Accuracy in order to determine the training accuracy of the model. The graph is shown in Fig 3.

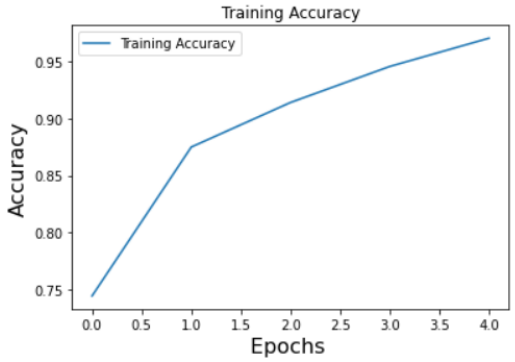

Fig. 3. Model Diagram

Similarly, In order to determine how well our model is trained we have plotted the graph of Epoch vs Training Loss as shown in Fig 4. It can be determined that till epoch 4 there is a significant decrease in the training loss but after that there is negligible change in the value. It can be clearly determined

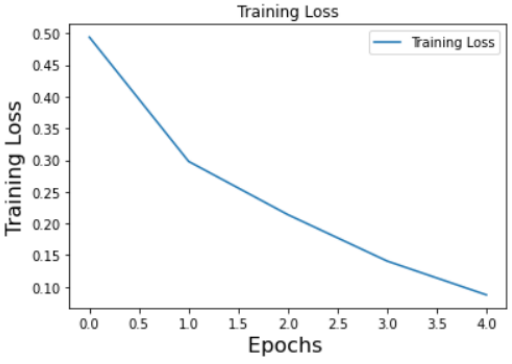

Fig. 4. Model Diagram

that the model get very well train in 5 epochs. And we received the training accuracy of more than 90% and loss went down to less than 0.15.

V. CONTRIBUTION

In order to implement this whole project as a team, we divided ourself into two groups for the coding purpose. Team 1 (Palak and Zalak) were working on the pre-processing steps after understanding the type of sentences which were there in the dataset. They went through the conceptual knowledge of all the pre-processing steps required for Natural Language Processing (NLP) and used the most convenient method for our project. Along with it team 1 implemented Support Vector Machine for classification in sentiment analysis. While Team 2 (Shiv and Pinak) started working using Convolution Neural Network (CNN) once the pre-processing steps were completed. We studied the different types of layers and its importance throughout NLP and developed the Convolution Neural Network model.

After successful implementation of both the variants we compared the results and found that using CNN model we are getting more efficiency so further we all started working on it to improve model performance. Once the Code was up and running, we reviewed our code and model in order to increase the accuracy by tuning the hyper-parameters. So, again team worked on the implementation of model. Both the team worked on improving the model but we kept the model which gave the higher accuracy. Model with the higher accuracy is used in the project while the model with less accuracy has been commented in the code. So, overall we all have done equal amount of work that is 25% of work done by each member in order to get 100% outcome. The whole project details including saved model, Report and code is available on the github and its link is this: https://github.com/PinakDivecha/NLP_Project_2

ACKNOWLEDGMENT

We as a team want to deeply thank our professor Dr. Thangarajah Akilan who helped us understand the concept of Natural Language Processing (NLP) and the machine learning. We also want to thank our TA's (Teaching Assistant's) Mr. Andrew and Mr. Punardeep who took keen interest and solved our issues and concerns regarding the code and error.

VI. CONCLUSION

In this project, we have used the Rotten movie review dataset containing more than 50,000 instances and five sentiment values. For the same, we have used the CNN model in order to predict the sentiment value based on the textual based movie review. We have performed proper pre-processing steps and converted the sentences into matrix format. We have used the matrix to build our CNN model using Pytorch framework and evaluated the model after doing proper hyper-parameter tuning. After hyper-parameter tuning, we were able to achieve the highest testing accuracy value of 85.86%.

REFERENCES

[1] Ramsha Ali, Shahzad Qasir, "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents" IEEE 2012

[2] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng and Christopher Potts, "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank" International Journal of Computer Applications, July 2018

[3] Sepp Hochreiter, Jaurgen Schmidhuber, "LONG SHORT-TERM MEMORY",Neural Computation, 1997

[4] Socher R, Lin C C, Manning C, et al, "Parsing natural scenes and natural language with recursive neural networks," In Proceedings of the 28th International Conference on Machine Learning (ICML), pp. 129-136, 2011.

[5] Socher R, Perelygin A, Wu J Y, et al, "Recursive deep models for semantic compositionality over a sentiment treebank," In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1631-1642, 2013.

[6] LeCun Y, Bottou L, Bengio Y, et al, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[7] Ciresan D, Meier U, Schmidhuber J, "Multi-column deep neural networks for image classification," In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, pp. 3642-3649, 2012.

[8] Ciresan D C, Meier U, Masci J, et al, "Flexible, high performance convolutional neural networks for image classification," In IJCAI ProceedingsInternational Joint Conference on Artificial Intelligence, vol. 22, no. 1,pp. 1237-1242, 2011

[9] Deng J, Dong W, Socher R, et al, "Imagenet: A large-scale hierarchical image database," In Computer Vision and Pattern Recognition (CVPR),2009 IEEE Conference on. IEEE, pp. 248-255, 2009.

[10] Kalchbrenner N, Grefenstette E, Blunsom P, "A convolutional neural network for modelling sentences," arXiv preprint arXiv:1404.2188, 2014.