Assignment 2 - Multi-class Sentiment Analysis using Convolution Neural Network

Arunim Garg
Department of Computer Science)
Lakehead University
Thunder Bay, ON
agarg27@lakeheadu.ca, #1092641

Abstract—**With the emergence of deep learning, convolution neural networks are being used to get fast and optimal results for various applications including natural language processing. This technical report delves into the architecture of a convolution neural network discusses the results of a customized convolution neural network with a tabular dataset for sentiment analysis.**

I. Introduction

In this report, a 1-d convolution nerual network (CNN) is used for a natural language processing (NLP) task of a multi- class sentiment analysis problem for a text-based movie review of Rotten Tomatoes avaiable at https://raw.githubusercontent.com/cacoderquan/Sentiment-Analysis-on-the-Rotten-Tomatoes-movie-review-dataset/master/train.tsv.

II. Background

Semantic Similarity Task (STS) is the task of examining and determining the degree of semantic similarity between two sentences [1]. STS is a really important task which can be considered as a building block for NLP tasks, because it helps determining the contextual meaning of the text with respect to a particular document in the corpus. Various vectorization methods like GloVe, word2vec etc., are used to convert the text into vectors of numbers in the pre-processing task before STS is incorporated.

Word2vec is a word embedding technique used to convert words in a text document to numerical vectors on which STS tasks can be conducted to find the similarity [2]. Word2vec uses different techniques like bag of words, TF-IDF, and n-grams to create the vectors. In a simple bag of words technique, the text is first ran through common pre-processing tasks like tokenization, removal of punctuation and stop words, stemming, and lemmization. Then, a vocabulary of all the words occurring in the document is created. Successively, the similarity of two sentences is then compared with the frequency of the words with respect to the vocabulary. The similarity measures used can be cosine similarity. TF-IDF also uses the same methodology but the creation of vocabulary is a bit different, since it also incorporates the frequency of words in one document vs the whole corpus.

III. Proposed Model

We propose to use a one-dimensional convolution (Conv1D)-based neural network to classify the phrases present in the Rotten Tomatoes dataset with its respective star rating (0 to 4). The sentiment is extracted from a sentence and matched with its respective sentiment score, i.e. the classes from 0, 1, 2, 3, 4 or 5. The classification model that we created included several convolution layers, several max-pooling layers, one flatten layer, and several linear layers for predicting the classes for the respective sentiment.

A. Methodologies

	Model details
Over/under fitting	train_test_split to shuffle the dataset
Vanishing/exploding gradient issue	ReLU and Batch normalization
No. of epochs	25
Learning rate	0.001
Optimizer	Adam
Activation Function	ReLU

TABLE I
Model details

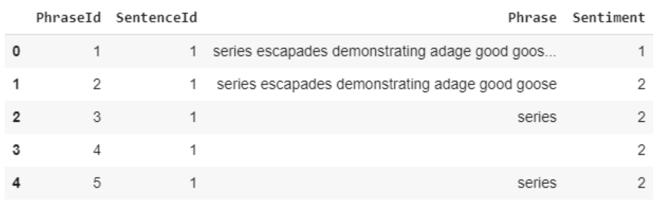
- To avoid over/under-fitting, the data has been shuffled and split into 70:30 for training and testing respectively. Hence, the testing data was not shown the model during the training phase, avoiding the issue of over-fitting.
- To avoid vanishing/exploding gradient issue, reLU activation and batch normalization is used.
- The kernel size is set 1x1 to extract as much data as possible in the tabular dataset.
- Number of epochs is set to be 25 to fully train the dataset. It was seen that the trend of loss and R^2 score would remain the same, if the number of epochs was increased. The loss and R^2 score is less with less number of epochs, therefore, 150 is set to be as the number of epochs with the most optimal scores.
- Learning rate is tuned to be 0.01. Different learning rates are measured with Adam optimizer, and 0.01 proved to be

the most efficient learning rate for this model and dataset, in comparison with 0.1 and 0.001 learning rates.

IV. Conclusion and results

The model is trained with 70% of the dataset and tested with the remaining 30%. The model's L1 loss is measured to be 42592 and R^2 is 0.737. The model used Adam optimizer with reLU layers for the convolution layers and linear layer for the linear regression.

In conclusion, the accuracy of this model was not high since 1-d CNN does not work well with textual tabular data, it would be optimal to work with a network like recurrent neural network or long short term memory network.



	PhraseId	SentenceId	Phrase	Sentiment
0	1	1	series escapades demonstrating adage good goos...	1
1	2	1	series escapades demonstrating adage good goose	2
2	3	1	series	2
3	4	1		2
4	5	1	series	2

Fig. 1. Head of the dataset

References

[1] Shao, Y. (2018). HCTI at SemEval-2017 Task 1: Use convolutional neural network to evaluate Semantic Textual Similarity. 130–133. https://doi.org/10.18653/v1/s17-2016

[2] Van Grinsven, M. J. J. P., Van Ginneken, B., Hoyng, C. B., Theelen, T., Sanchez, C. I. (2016). Fast Convolutional Neural Network Training Using Selective Data Sampling: Application to Hemorrhage Detection in Color Fundus Images. IEEE Transactions on Medical Imaging, 35(5),1273-1284.

Appendix