# PREDICTING YELP'S TEXT REVIEWS' STAR RATINGS USING CONVOLUTIONAL NEURAL NETWORKS

**DV 2542 - MACHINE LEARNING**
**PROJECT (5.5 ECTS)**

K S SUSHEEL SAGAR
9303177837
suko15@student.bth.se

## I. INTRODUCTION

Yelp is an online urban guide that helps people find recommended destinations based on the informed opinions and reviews of a community. It is one of the largest user review websites, that hosts an online database of user generated reviews of local businesses. Every year, yelp organizes a data science challenge and provides the data science enthusiasts with the real review data, generated by the yelp users, and also the challenging problems of interest in various domains such as Social Graph Mining, Natural Language Processing, Location Mining, Urban Planning etc. One of the problems in the latest data science challenge, under the Natural Language Processing (NLP) domain is - *" How well can you guess a review's star rating from its text alone? "* [1].

The process of analysing the text to determine the feelings expressed towards the entities, events and their properties is called as sentiment analysis [2]. In the past decades, sentiment analysis was done using the traditional classification models such as Naive Bayes (NB), Naive Bayes with bag of bigram features (BiNB) and support vector machines (SVM). Recently, sentiment analysis using deep learning models has been proven to be exhibiting better performance than the traditional models. However, deep learning models have very high computational complexity, compared to the traditional classification models [3].

## II. RELATED WORK

Sentiment analysis of online user generated content is one of the most researched areas within the natural language processing domain. Among the several approaches proposed to solve the sentiment analysis problem, bag of words technique was one of the most widely used one [4]. Apart from this, google proposed an open source tool called Word2vec which uses the Continuous Bag Of Words (CBOW) model. Word2Vec creates features without human intervention. It takes text corpus as input and produces the word vectors as output. It first constructs a vocabulary from the training text data and then learns vector representations of words. The resulting word vectors can be used as features in many natural language processing tasks. In the article [5], by google, the authors clearly explain that the CBOW model is better than the Bag of words.

With the tremendous success of deep learning models in the supervised and unsupervised learning fields, several researchers are trying to solve the sentiment analysis task using deep learning algorithms. One of the famous deep learning framework for sentiment analysis was the Recursive Neural Network (RNN) proposed by Socher[6]. This framework used a fully labeled parse trees to represent movie reviews from *rottentomatoes.com.* Socher later proposed an improved recursive neural network called Recursive Neural Tensor Network (RNTN) in which, a tensor based composite function for all the nodes in the parse trees. However Ouyang et al. argue that both the approaches proposed by socher require a fully labelled parse trees to train the neural network and it is computationally costly. As an alternate solution to this problem, they proposed a convolutional neural network (CNN) that only requires the labels of the full sentences (a single review). This CNN has much fewer connections and parameters, hence making it easier to train, compared to the Recursive networks proposed by socher. Additionally, they have proposed a framework for sentiment analysis using word2vec and CNN, and have achieved better accuracy than the state of the art implementations such as RNN, RNTN, Matrix vector RNN and the traditional methods like NB, BiNB and SVM [3].

## III. PROJECT GOALS

**Goal 1 :** To predict the star ratings of yelp's text reviews using word2vec and convolutional neural network framework proposed in [3].
**Motivation :** From the literature, the framework proposed by Ouyang et al. is proven to have a very high accuracy compared to other deep learning models and traditional classification models [3]. Ouyang et al. also argue that their framework is much simpler to train, compared to other deep learning models such as RNN, RNTN etc. For these reasons I have chosen this framework (the combination of word2vec and convolutional neural network) for the task of predicting the star ratings of yelp reviews from their text alone.
**Non Standard Task** : Deep Learning.

**Goal 2 :** To compare the training time of this CNN model on a CPU and a GP-GPU.
**Motivation :** In their paper Ouyang et al. have conducted their experiments on a GPU. How ever they did not mention the exact training time of the neural network. They just

mentioned that the training time was very low on a high end GPU. Research on implementing CNNs on GPU show that the performance of GPU based implementations have achieved a better speedup over the CPU based implementations [7][8]. For this purpose I would like to compare the performance of this CNN (proposed in [3]) on a CPU and a GP-GPU.
**Non Standard Task** : GPU based learning.

## IV. PROBLEM FORMULATION

Each review posted in yelp, will have a short text description about the product or service, and also the star rating given to it. The star rating ranges from 1-5, where 1 stands for very bad, 2 for bad, 3 for good, 4 for very good and 5 for Excellent. In this project I consider the rating as a class label for the text review. Hence in the current problem there are 5 classes 1,2,3,4 and 5 that correspond to the rating of a text review. Hence this problem can be considered as a Multi Class Classification problem.

## V. EXPERIMENTATION

### A. Dataset :

The data set used for the experiments is provided by the yelp data science challenge 2016 [1]. It consists of 2,225,213 (2.22 million) reviews. Each review consists of the following fields :
i. user_id : A unique identifier of a yelp user.
ii. review_id : A unique identifier for each yelp review.
iii. business_id : A unique identifier for each business.
iv. Star : The star rating given to the business by a user.
v. Text : The text review given by the user, explaining his/her experience with the product or service provided by the business (corresponding to the business_id).

However, for this project, due to limitations with the computational capabilities, only 24,000 reviews were considered for the experimentation. As the model proposed in [3] has demonstrated a high accuracy on the reviews with the number of words being less than 54, reviews that contains less than 54 words were selected for the experiments. Additionally, majority of this data set ( with 2.2 million reviews) consists of reviews less than 54 words. For these reasons, this decision has been made. This has been discussed in a greater detail in section VII (Limitations and Future work).

### B. Word2Vec :

As proposed in [9] word2vec can implement two types of neural network models called Continuous Bag of Words (CBOW) Model and Continuous skip gram model. Both these models outperformed other standard models such as bag of words. Using one of these models word2vec can make a highly accurate guess about a word's meaning. For this project I followed the methodology used in [3] for generating the word vectors. I have used the pre trained vectors trained on Google news data set with about 100 billion words. The model can be freely downloaded from [10]. It contains 300 dimensional vectors for 3 million words and phrases. For the words which did not appear in these 3 million words, using the online learning capability provided by gensim's word2vec, the 300 dimensional vectors were generated [11].

The input data of the CNN is expected of the same size, which means all the input to CNN will have same dimension. As proposed in [3] for the reviews with less than 53 words, zero vectors will be added at the end of the sentences. Hence all the reviews input given to CNN will be 53*300 dimensional.

**C. CNN :**

As the main goal of this project is to implement the CNN proposed by Ouyang et al. for predicting the star ratings of yelp reviews, all the layers are arranged in the same manner and all the parameters are set to the same values as in [3]. The neural network consists of 3 convolutional layers and 3 pooling layers. Fig 1 from [3] presented below explains the CNN architecture used in this project.

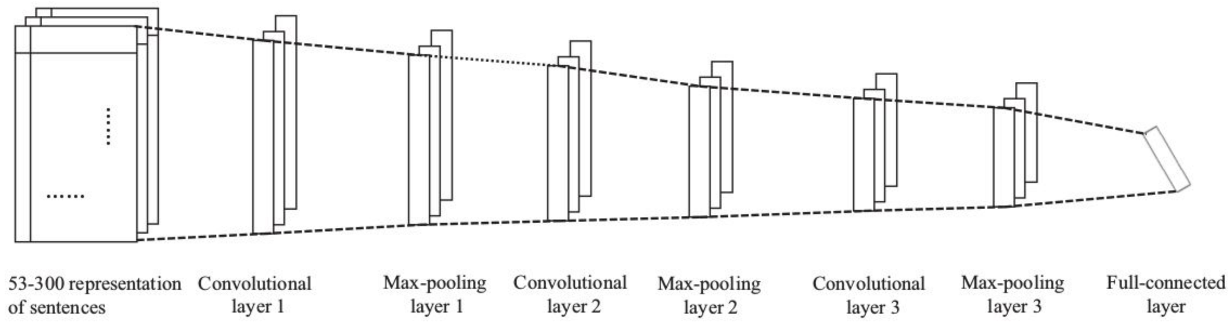

Fig. 1: The architecture of our CNN. The size of network's input is 53×300.

The input is processed by three convolutional layers. The first two convolutional layers are also followed by Parametric Rectified Linear Unit layers (PReLU), dropout layers, max-pooling layers and normalization layers. The first convolutional layer has 50 kernels of size 4 X 5 with a stride of 5. Then there is a PReLU layer to improve model fitting. In the dropout layer, the dropout regularization ratio is set to 0.5 to reduce complex adaptations of neurons in the nets. Then there is a max pooling layer whose kernel size is 2 X 2 and stride is 2. The second convolutional layer has 100 kernels of size 5 X 6. Similarly, there are PReLU, dropout, max pooling and normalization layers after the second convolutional layer. After these layers there is a last pair of convolutional and max pooling layer. Furthermore there is only one fully connected layer in the net. The neural network's size and parameters are given in the below table.

| Input Size | No. of Hidden layers | No. of Output Labels | Iterations | Momentum | Base Learning Rate | Gamma | Step Size |
|---|---|---|---|---|---|---|---|
| 53 X 300 | 6 | 5 | 14500 | 0.9 | 0.001 | 0.1 | 800 |

Table 1 : Parameters for the CNN

**D. Softwares used and their application :**

In this project, the following software packages are used:

**i. NLTK :** NLTK is a natural language toolkit for python programmers, which provides a suite of text processing libraries for classification, tokenization, stemming, parts of speech tagging etc. In this project, NLTK is used to remove the punctuation marks such as ',' '.' etc., from the text reviews. Additionally, it is also used for tokenization of sentences i.e., to break the sentences into individual words [12]. In the source code, the file 'review_read.py' implements this functionality.

**ii. Gensim :** It is a NLP api that contains several robust, efficient and hassle free piece of software to realize semantic modelling from plain text [11]. Gensim provides a python implementation of google's word2vec (originally written in C), along with some additional functionalities such as online learning capability, that allows the programmers to load a previously trained model and continue training it with more sentences. In this project, Gensim is used to generate the vector representations of the reviews' words generated from the tokenization function of NLTK. In the source code, the file 'word_to_vec.py' implements this functionality.

**iii. Numpy and H5py :** The vector representations obtained from gensim are stored in the form of numpy arrays. H5py is a python api to convert the data in the numpy arrays into HDF5 format which is one of the standard input format accepted by Caffe. In the source code, the file 'pkl_to_hdf5.py' implements this functionality.

**iv. Caffe :** Caffe is one of the most widely used deep learning framework made with expression, speed, and modularity in mind [13]. For these reasons, I have chosen Caffe to implement the CNN [3]. It supports both CPU and GPU implementations of various state of the art deep learning algorithms. Caffe accepts input in the form of HDF5, LMDB and LevelDB. The attached folders ‘conv5’ and ‘conv6’ in caffe_files folder contains all the necessary CNN files.

As the HDF5 files are of size greater than 1.4 GB, these files are not attached in the submission.

### E. Hardware :

All the experiments are conducted on a laptop PC with Intel core I7-4710MQ octacore processor with a clock rate of 2.5GHZ and 16 GB RAM. The laptop also has a dedicated Nvidia GTX 765M GPU with 768 cores each with a clock frequency of 850 MHz.

## VI. RESULTS AND ANALYSIS

The results of the experiments conducted to realize goal 1 and goal 2 are presented in the below table (Table 2).

| Training Data (No. of Reviews) | Test Data (No. of Reviews) | Iterations | Accuracy | Learning Time On CPU (Minutes) | Learning Time on GPU (Minutes) |
|---|---|---|---|---|---|
| 10000 | 2000 | 14500 | 0.3471 | 99 | 25 |
| 20000 | 4000 | 14500 | 0.5566 | 96 | 61 |

Table 2 : Results of Experiments

**Results corresponding to Goal 1 :**

On the training data of 10000 reviews, the accuracy of the CNN is 0.3471 which is lower than the accuracy obtained in [3] i.e 0.454 on a training set of 9645 reviews. I think this is because the training set used in [3] uses the reviews of the single domain i.e hollywood movies. Whereas, the data set used in this project consists of reviews of different businesses such as restaurants, spas, automotives etc. However when the training data has 20000 reviews the accuracy of the system increased to 0.5566. This might be because of the increase in dataset size, which helps in learning a better model by the business domain adaptation of the CNN. This means, different positive words of various business domains, such as ‘delicious’ (used in case of restaurants), ‘reliable’ (used in case of automotives), ‘beautiful’ (used in case of spa) are learnt by the neural network to predict the respective classes in an effective manner. The screenshots of the experiments after 14500 iterations are attached in the screenshots folder of this submission.

**Results corresponding to Goal 2 :**

The learning time of the CNN model on a CPU for the training set of 10,000 and 20,000 reviews is 99 minutes and 96 minutes respectively. That is, the training time for 20,000 reviews is less than that for 10,000 reviews. This might be because of the octacore CPU that has been used for the experiments. In case of 10000 reviews, the communication overhead among the cores might be the cause for a high learning time, which in case of 20000 reviews might be less and thus resulting in a low training time.

However, in both the cases, the training time on GPU is less than that of CPU. In case of the training set of 10,000 reviews, the speedup attained on a GPU is 3.96 and on 20,000 reviews, it is 1.573.

## VII. LIMITATIONS AND FUTURE WORK

The major limitation of this project is that, only the reviews with less than 54 words are considered for experimentation, as the CNN in [3] exhibited highest accuracy with the vectors of form 53*300. However, to obtain a generalized solution, additional experiments are to be conducted on the reviews with any number of words i.e on the reviews containing greater than 53 words. Apart from this limitation, during the conversion of the numpy arrays to HDF5 format, as explained in section V.D.iii , as all the data in the form of numpy arrays (with floating point values) has to be loaded into memory, for datasets larger than 20000 reviews, I have faced memory leak problems, which caused the program to terminate intermediately. For this reason, I could not obtain the HDF5 formats of large number (greater than 20000) of reviews and hence the training data is limited to 20000 reviews. With higher computational capacity (especially RAM) larger data can be converted to HDF5 format and can be given as input to CNN.

I think that with the larger data sets (more number of reviews) the accuracy of the CNN might also increase or at least does not decrease than that of the 20000 reviews' case. The speedup attained from that of the GPU has to be examined with the larger data sets, as the present results are not sufficient enough to comment about the generalised speedup. However, it can be stated that the learning time of this CNN is less on a GPU, compared to that on a CPU.

**References**

[1] https://www.yelp.com/dataset_challenge

[2] Raghupathi, Dilip, Bernard Yannou, Romain Farel, and Emilie Poirson. "Customer sentiment appraisal from user-generated product reviews: a domain independent heuristic algorithm." *International Journal on Interactive Design and Manufacturing (IJIDeM)* 9, no. 3 (2015): 201-211.

[3] Ouyang, Xi, Pan Zhou, Cheng Hua Li, and Lijun Liu. "Sentiment Analysis Using Convolutional Neural Network." In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, pp. 2359-2364. IEEE, 2015.

[4] Pang, Bo, and Lillian Lee. "Opinion mining and sentiment analysis."*Foundations and trends in information retrieval* 2, no. 1-2 (2008): 1-135.

[5] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

[6] Socher, Richard, Cliff C. Lin, Chris Manning, and Andrew Y. Ng. "Parsing natural scenes and natural language with recursive neural networks." In*Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 129-136. 2011.

[7] Izotov, P. Yu, N. L. Kazanskiy, D. L. Golovashkin, and S. V. Sukhanov. "CUDA-enabled implementation of a neural network algorithm for handwritten digit recognition." *Optical Memory and Neural Networks* 20, no. 2 (2011): 98-106.

[8] Liu, Qi, Zi Huang, and Fuqiao Hu. "Accelerating convolution-based detection model on GPU." In *Estimation, Detection and Information Fusion (ICEDIF), 2015 International Conference on*, pp. 61-66. IEEE, 2015.

[9] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

[10] https://code.google.com/archive/p/word2vec/

[11] https://radimrehurek.com/gensim/

[12] http://www.nltk.org/

[13] caffe.berkeleyvision.org