Sentiment Classification of Yelp reviews: A Neural Network Approach

Kailash Nathan Srinivasan, Neelesh Kumar Boddu

Abstract

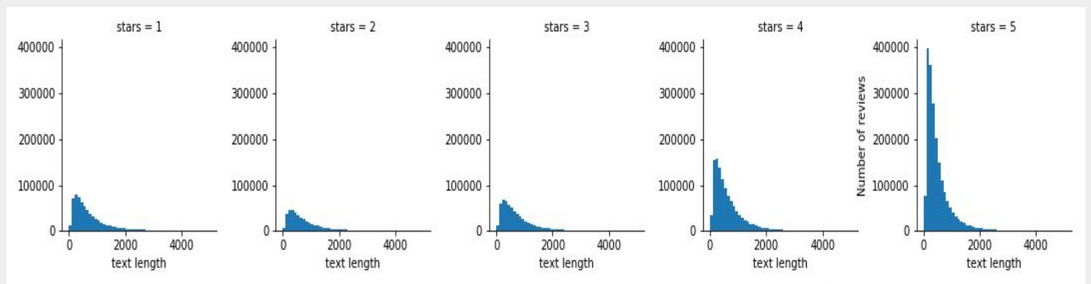
Neural networks, especially CNNs have recently achieved remarkable performance in the NLP task of sentiment classification, mainly due to their ability to learn sequential and semantic relationships. In this project we explore various neural network approaches to sentiment analysis of Yelp reviews using the 'Yelp Challenge dataset'. Specifically, we aim to use the review text to predict the star rating for the business, which is a quantitative measure of positive or negative sentiment.

Data

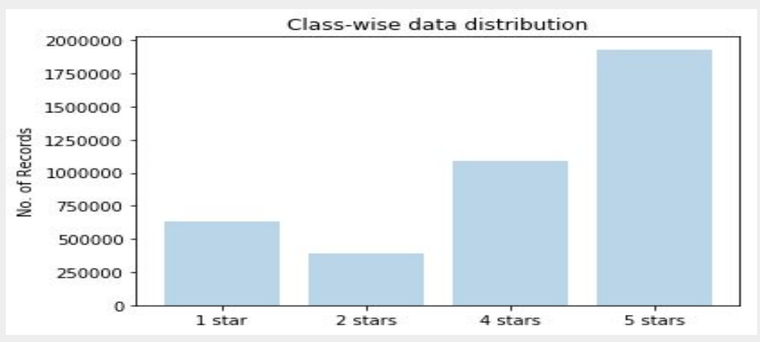
The dataset we use is the Yelp 2017 challenge dataset,introduced in the 10th round of Yelp Challenge,comprises user reviews about local businesses in 11 cities across 4 countries with star rating from 1 to 5. The large-scale dataset comprises 4.7M reviews and 947 K tips by 1M users for 144K

Data exploration & pre-processing

We filter out non-English reviews and examine review text lengths.



We filter out reviews with 3 stars to help us polarize the dataset. We then plot the class-wise distribution and see that the data is significantly skewed.



This helped us make sense of the most common class baseline predictions and guide us in balancing the dataset. We then balance and clean the review texts from the balanced dataset. Our cleaning process involves removing HTML tags(if any), random characters(incl. some punctuations), english stopwords, stripping extra white spaces and changing to lower-case.

We first select a part of the dataset, perform the balancing and cleaning, and use the text lengths and word counts from this new to help us get started with a range for the max_text_length and max_num_words. Using this dataset and these parameters, we proceed to training models on the data.

Models We Tried

Baselines

Our baseline models and accuracies were as follows:

	UNBALANCED DATA	BALANCED DATA
MULTINOMIAL NAIVE BAYES	58	55
RANDOM FOREST	61	59
MULTI LAYER PERCEPTRON	66	64

Word2Vec

Word2vec is a way of representing words and phrases as vectors in medium dimensional space. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words.We have used word vectors with deep learning via word2vec's "skip-gram and CBOW models", using either hierarchical softmax

Downsampling rate: 1e-3	Context window: 10
Number of features : 128	Number of threads in parallel: 4

Neural network training

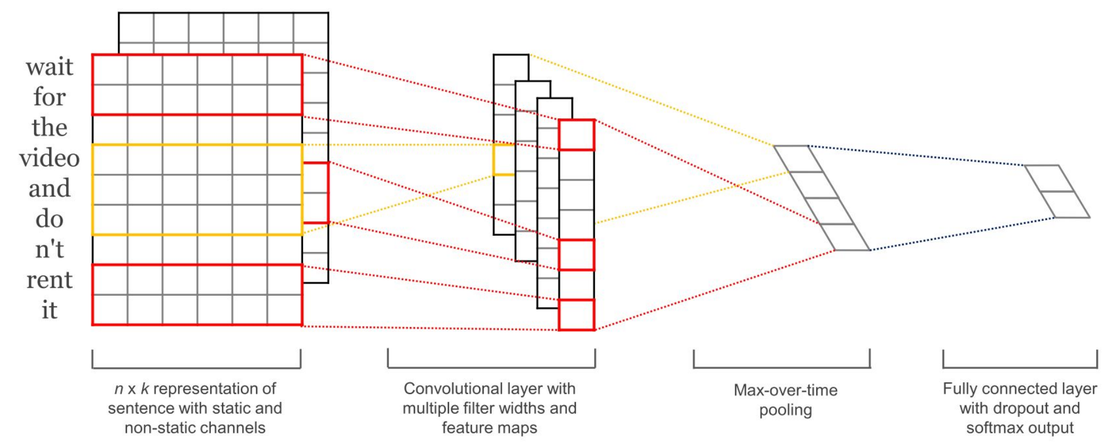
Problem: Training neural networks takes a long time and requires a lot of memory.
Solution: Use GPU!!!
Our implementation:
We configured our systems to use Tensorflow GPU version
To take advantage of the 4 GB NVIDIA GTX 960M gpus
This involved configuring the CUDA toolkit and CUDNN

MultiLayer Perceptron

Our MLP model has 3 hidden layers of 128, 64 and 32 units with Relu activations. After Hyper-parameter tuning, we settled with these hyperparameters:

Learning rate : 1e-5	Word2Vec dimensions: 128
Hidden Layer Dimensions: 128, 64, 32	Optimizer: SGD

Convolutional Neural Network Model

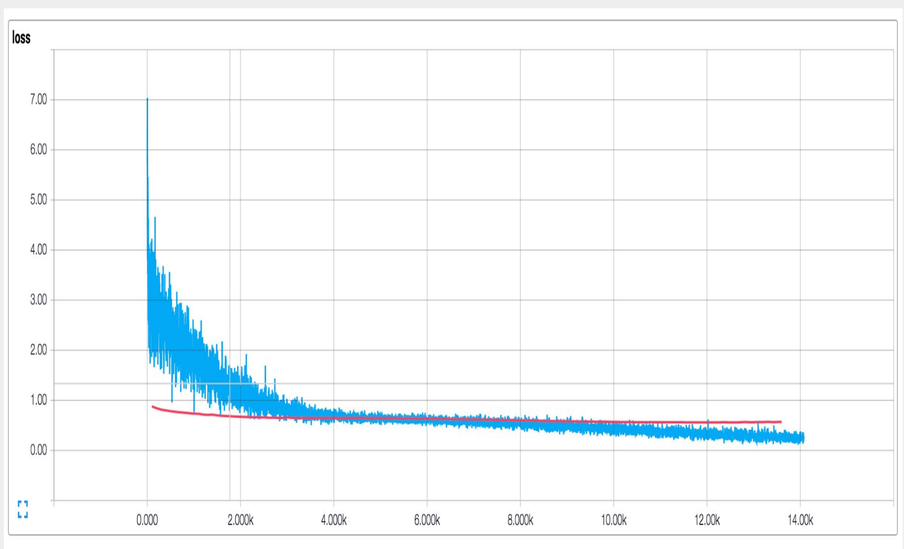


Embedding Dimension:128	Dropout Probability: 0.5
No of filters:128	Optimizer: Adam

Results Obtained:

From our experiments, we learnt that CNN performed better than any other methods we tried.For a binary classification where we tagged ratings >4 as Positive and less than that as Negative
Accuracy obtained on balanced data for binary classification : ~ 69%.
Accuracy obtained on unbalanced data for binary classification : ~73%

Loss plot:




Class-wise data distribution
No. of Records
2000000
1750000
1500000
1250000
1000000
750000
500000
250000
0
1 star
2 stars
4 stars
5 stars