Neural Networks final project:

Amazon digital music reviews sentiment analysis

Abstract

Sentiment analysis is one of the hottest research field in natural language processing. Therefore, I would like to implement both Convolutional Neuron Networks and Multiple Layer Perceptron on sentiment analysis in this research. The most common problem in sentiment analysis by machine learning is word embeddings from data set. Hence, my target is to identify which type of words presentation is the most suitable and lead to the highest accuracy in the problem of classify negative or positive reviews from Amazon dataset.

Introduction

Through many centuries, human being uses languages to communicate with each other so that their ideas, opinions, emotions, attitudes, sentiments can be expressed. Because of the existence of social media, people can easily express their opinions in Facebook, Twitter, Youtube, goods reviews in the Web etc. that leads the evolution of sentiment analysis which focuses on objects and subjects' behavior and information. So how would researchers and developers present human opinions as data for machine learning? The problem is hard because the system of communication in human society is extremely complicated such as grammar, metaphor, etc. The appropriate strategies for researchers are to use statistical methods such as "bag of words", support vector machines to present words in numbers and vector forms. The project would do some experiments in design and implementation the use of different statistical methods to present words from the reviews in the dataset such that the data can be used to fit in machine learning algorithms. Hence, in this research I want to study the problem of presenting data so that the machine would be able to efficiently learn the patterns of digital music reviews from Amazon customers.

1. Background

Dataset

The dataset includes metadata and product reviews from Amazon in Json format [1] [2]. In particular, the dataset includes reviews such as rating, review text, etc. There are 64706 reviews in total, which is about digital music of different music artists with rating from 1 to 5. However, the project modifies the dataset becoming 10920 reviews labeled by pure positive (5 starts only) and negative (1 – 2 starts) reviews.

Table 1 The table lists the number of reviews for each start rating in Amazon digital music data set.

Rating start	Number of ratings
☺ ☺ ☺ ☺ ☺	35580
☺ ☺ ☺ ☺	16535
☺ ☺ ☺	6789
☺ ☺	2791
☺	3010

The dictionary (the total number of words) for the dataset is 107680 words, which is very large. How the dataset has been modified from 64706 reviews down to 10920 reviews will be explained in detail on the part of methodology.

Libraries

There will be 3 main libraries will be used for design and implementation of machine learning:

Keras

Keras library is a high-level API provided by TensorFlow to build and train deep learning models. Keras is popular and be used widely because it is simple and easy to learn. In the project, Keras is used to build 1-dimension convolution neuron network.

Scikit-learn

Like Keras, Scikit-learn is a machine learning library that supports different tools to implement, preprocess data, select model etc. Scikit-learn support an efficient implementation of K-Folds cross-validator, which will be used in the project to evaluate the performance of deep learning models.

TensorFlow

Google's TensorFlow is one the most popular open source machine learning platform. The platform is especially designed for efficient computations, models visualizations in deep learning. A multi-layer perceptron will be designed and implemented by using TensorFlow library.

2. Problem Statement

The goal of the project is to be able to classify reviews are whether positive or negative with the high accuracy. From three different levels of sentiment analysis, the level of difficulty of the project is document level, which is also known as *document-level sentiment classification* [3]. To approach to the solution for the problem, the machine should be able to learn feature-based opinions from reviews' text. However, there are many problems and obstacles along the way to approach the goal.

First, human beings have a very high level of abstraction, therefore, the language being used by them are extremely complex. According to Liu [3], sentiment lexicon is not sufficient for sentiment analysis, but it is necessary for example: a negative or positive sentiment word can have both orientations depending on scenarios. Or many sentences do not contain any sentiment words but still owning opinions such as "*the machine requires a lot of energy*".

In addition to that, machine learning algorithms requires fit data to be presented as a fixed vector or matrix. So how I would be able to convert text review into vector? To meet the need of the presenting data for machine learning algorithms, one potential solution is Word Embedding which tends to present texts by numbers. Two most common text categorization methods in Word Embedding is to present all words in vector based on a given dictionary: bag of words and bag of n grams. I would test the implementation of the bag of word model on dataset in deep learning models CNN and MLP. Hence, I expect that the machines would be able to learn the patterns to achieve a high classification accuracy rate.

3. Related Work

The research of Bespalov et al. [4] on sentiment classification. In details, they design and implement a new embedding mechanism, called "latent n-grams" based on n-grams to modify their datasets. Thereby, the machine would be able to learn and predict the expressed opinions in the text where it is positive, negative and the Likert-scale. According to Bespalov et al. [4] "latent n-grams" would greatly reduce the dimensionality of n-grams and the main sentiment classification task will be optimized learned using supervised signals. My work is different than their work at some points: My current work only contains only one dataset and my new methodology is much simpler version compared to theirs new embedding mechanism. In addition, the Liker-scale will not be involved in my work instead I only aim at sentiment analysis on textual contents so that machine would be able to identify positive and negative opinion.

In Hu and Liu's research [5], they also work on customer reviews but their task aims on mining product features, indicating opinion in each sentences in reviews. Their work is more specific compared to my work and Bespalov et al' work [4]. Hu and Liu's work is at the hardest level of sentiment analysis which is known as "entity and aspect level" [3]. The goal of the level is to carefully discover sentiment of the whole sentences in details such as: "The sound quality is great, but the headphone is heavy to me". At the level, two aspects: sound quality and headphone's weight will be evaluated. In details, they use WordNet [6], which is a large vocabulary database of English to support the search method in semantic orientations prediction. My research goal is just on document level which is more vague and much simpler than the research done by Hu and Liu [5].

My work does use WordNet but just only use for mining adjective words from the WordNet's dictionary rather than performing complex algorithm. The research is also related on sentiment classification on customer reviews by CNN and MLP. More details are discussed carefully below.

4. Methodology

Transforming the dataset

Unlike other dataset, reviews dataset has textual content, hence, the data set contains punctuations, brackets, dot, caret, question (?), comma, etc. Therefore, the dataset must be modified to remove any unnecessary characters. Thanks to regular expression operations (re) library, the library supports replace function to remove unnecessary characters to make the dataset becomes more clean and easier to be analyzed.

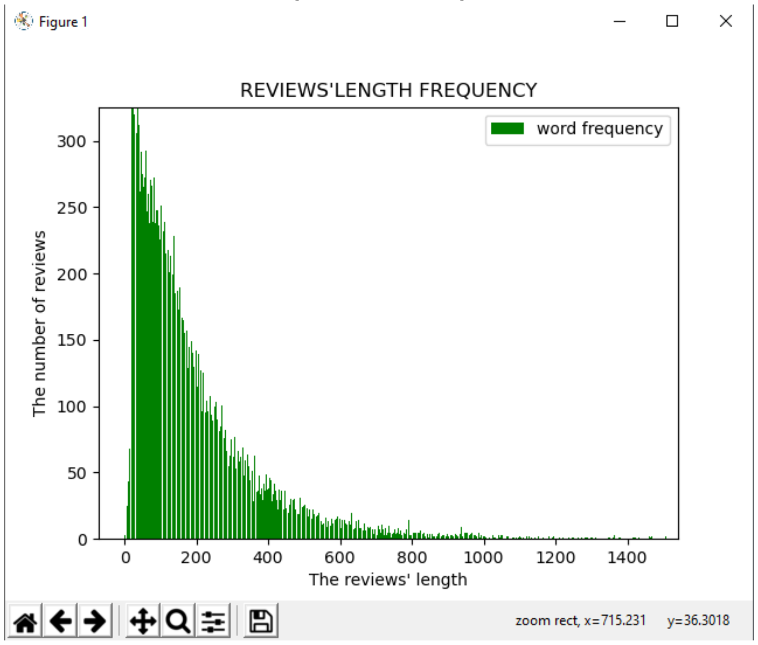

Figure 1: The attribution of length of word in reviews of the dataset

After the texts in the dataset is processed, then it will be modified so that the data only contains positive and negative reviews. I assume that 5 starts are positive reviews, 3-4 starts are neutral reviews, and 1-2 starts are negative reviews. From the table 1, I can see that there are only 5801 negative reviews and 35580 positive reviews. I would want to convert text into vector, but the size of reviews is a serious problem.

The dataset has 107680 different words in total, this statistical is generated by Keras library. That is hard to present a vector has length 107680 as the input for the deep learning network. From the Figure 1, I see the attribution of reviews' length of the dataset. The graph is skewed to the right, in other words, most of the reviews has length of 400 or less. As the result, the project would only choose review has the length of 400 or less. After modifying the data, overall, the dataset contains only have 37950 words in total and the dataset contains 10920 reviews only. Along with that, most negative reviews have less than 400 words, hence, 400 is an appropriate number to modify the dataset.

Table 2: The dataset after modifying.

Sentiment	Number of reviews
Positive	5460
Negative	5460

Presenting data

The research would use a multiple layer perceptron and convolution neuron network model for sentiment analyses; therefore, I need appropriate ways to present texts so that deep learning neuron network can learn the patterns from the datasets.

The text in dataset will be presented by three different ways:

1. For the first method, the text in dataset will be presented as vector based on the word frequency of the data set. In other words, I update internal vocabulary based on the dictionary of the dataset but the most 8000 common words will be kept. The method will be done by Tokenizer a class provided by Keras to vectorize text. In addition, Tokenizer class will automatically identify which words are related or closed to other to convert words into numbers. The algorithm behind the class might be extremely complex.
2. Second method is the text in dataset will be vectorized based on the adjective dictionary of the dataset. The collection of adjective words from the dataset will be done by using WordNet [6]. In total, the dataset has 9537 adjective words, then I update internal vocabulary based on the 9537 adjective words in tokenizer class. Next, I convert reviews by text to sequence function in Tokenizer class.
3. Final, I use global vector for word representation, GloVe, to present a word as a vector [7]. GloVe provides a set of data to present a word as a vector size of 100. As the result, a review can be presented as a matrix size of [400,100], I use padding for reviews has less than 400 words.

Model
I use 20% of the modified dataset for validation and 80% of the modified dataset for training. As mentioned above, there will be 2 models used in the research MLP and CNN. In general, multiple layer perception has only 2 hidden layers. There will be two different MLP, because there are two ways to present data for fitting data in MLP.

MLP uses RMS with learning rate 0.01. The activation in the model is sigmoid and the most appropriate standard deviation and drop out rate is respectively 0.15 and 0.5 through some quick experiments.

The model for the first strategy:

- Input layer (8000)
- First hidden layer (8000, 625)
- Second hidden layer (625,300)
- Softmax (300,2)

The model for the second strategy:

- Input layer (9537)
- First hidden layer (9537, 625)
- Second hidden layer (625,300)

- Softmax (300,2)

The model for the CNN, because the input is a matrix [400,100], therefore, I have to convolution for one dimension. In addition, the activation used in the model is relu, the kernel size use in the model is: [10x100]; pooling size of 3.

Layer (type)	Output shape	Parameter
Conv1d	None, 391, 100	100100
Conv1d_1	None, 381, 100	100100
Max_polling1d	None, 127, 100	0
Conv1d_2	None,117, 100	110100
Max_pooling1d_1	None, 39,100	0
Flatten	None, 3900	0
Dense	None, 625	2438125
Dense_1	None,2	1252

5. Results

Table 3: The results from 3 methods

Method	Accuracy
Tokenizer: word frequency	92.26%
Tokenizer: adjective words	50.78%
Tokenizer: adjective words (without updating dictionary)	88.18%
Word embedding: GloVe	87.36%

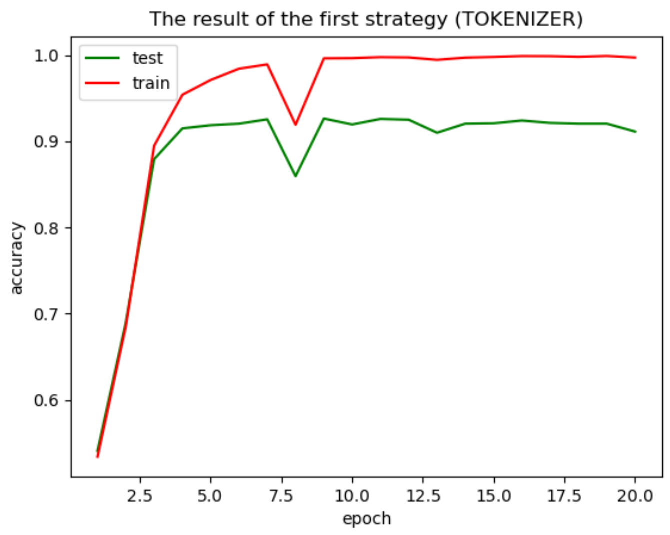

Figure 2: The result of the first method. Using word frequency in Tokenizer class

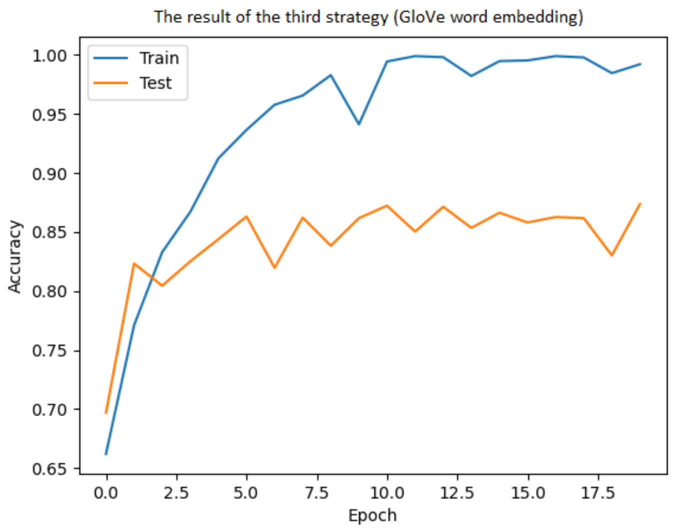

Figure 3: The result of third method. Using word embedding GloVe

From the Figure 2, I can see that at fifth epoch, the model is overfitting the data which is not well enough even though the accuracy is up to 92%, however, the model is overfitting on the training set. The easy way to recognize this is the performance difference between train and test lines in the graph.

Like the first strategy, the CNN model is soon overfitting on the training set at epoch 6. The accuracy of the CNN is 87.36% which is not good as the first strategy, the word frequency provided by Tokenizer class. Let's look on the performance of the new method: fitting the adjectives word strategy. Below is the comparison between without fitting and fitting the adjective words into the internal dictionary of the dataset.

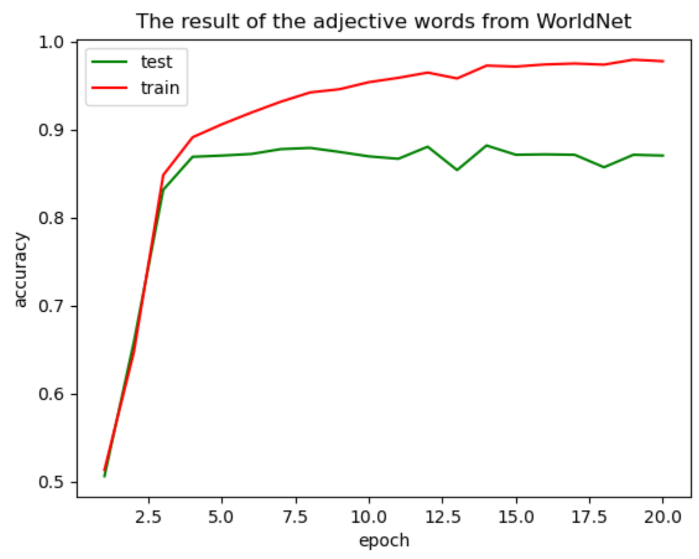

Figure 4: fitting the dictionary of the dataset on Tokenizer

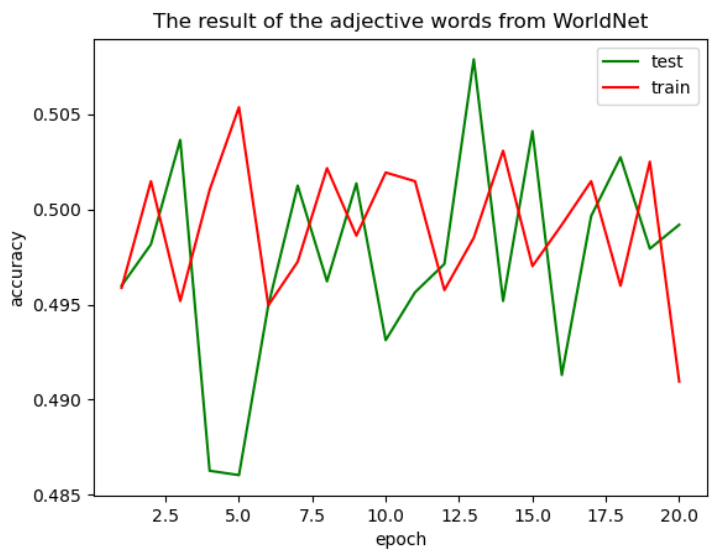

Figure 5: Without fitting the dictionary of the data on Tokenizer

We can see the significant difference between fitting and without fitting the dictionary of the dataset on Tokenizer. It strongly affects on the process of vectorization texts of reviews. The fitting dictionary has a higher accuracy compare to without fitting the dictionary. From that we can see the important of fitting words into dictionary in Tokenizer class. The left-hand side is the Figure 5, which summarizes the performance of three methods.

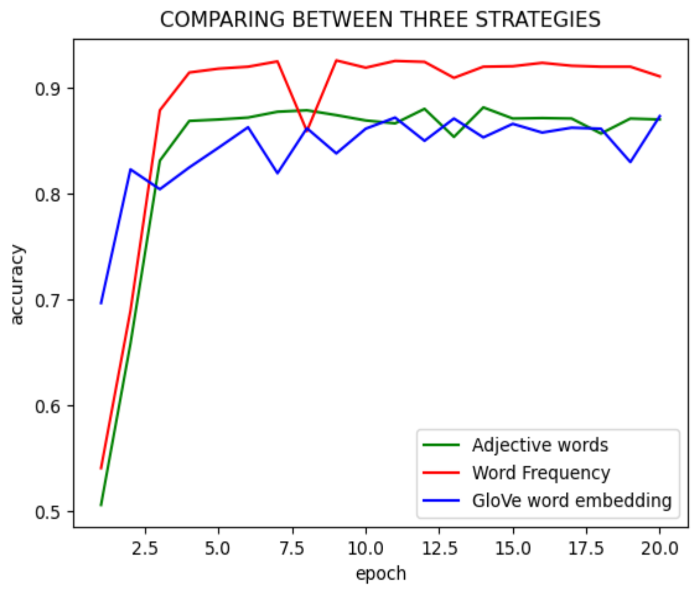

Figure 6: Summary of three methods

6. Conclusion and Future Work

As the definition of deep learning, which is part of machine learning methods based on learning data representation. In this experiment, the Tokenizer word frequency is suitable and efficient for the experiment. Because the deep learning model can classify with high accuracy within few epochs. If the word embedding is more improved the model would perform better. The success of the method is perhaps behind the Tokenizer class, it has already covered many advantage algorithms for word embedding better than GloVe or the new method I just created.

The research would be useful for relevant sentiment analysis even though the current level of the research is very simple which is “document level”. In addition, the success of implementation in adject words indicates that Liu [3] is right to claim that sentiment lexicon is not sufficient for sentiment analysis, but it is necessary. In other dataset or the requirement in the dataset is more specific, adjective words might not enough to achieve the accuracy. Overall, in the experiment, the project successfully indicates that the implementation of word frequency in Tokenizer to vectorize the dataset would help the deep learning network successfully achieve 92% accuracy.

References

[1] R. He and J. McAuley, Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, WWW, 2016. Available: https://dl.acm.org/doi/10.1145/2872427.2883037 [Accessed April 9, 2020]

[2] J. McAuley, C. Targett, J. Shi and A. v. d. Hengel, Image-based recommendations on styles and substitutes, SIGIR, 2015. Available: https://dl.acm.org/doi/10.1145/2766462.2767755 [Accessed April 3, 2020]

[3] B. Liu, Sentiment Analysis and Opinion Mining, Morgan & Claypool, 2012. Available: https://www.cs.uic.edu/~liub/FBS/SentimentAnalysis-and-OpinionMining.pdf [Accessed April 5, 2020]

[4] D. Bespalov, B. Bai, Y. Qi and A. Shokoufandeh, "Sentiment Classification Based on Supervised Latent N-Gram Analysis," *Proceedings of the 20th ACM International Conference on Information and Knowledge Management,* vol. 8, pp. 375-382, 2011. Available: https://dl.acm.org/doi/10.1145/2063576.2063635 [Accessed April 7, 2020]

[5] B. Liu and M. Hu, "Mining and summarizing customer reviews," *In Proceeding of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004),* vol. KDD, no. 04, pp. 168-177, 2004. Available: https://dl.acm.org/doi/10.1145/1014052.1014073 [Accessed April 9, 2020]

[6] C. Fellbaum, "WordNet: An Electronic Lexical database," MA: MIT Press, Cambridge, 1998. https://wordnet.princeton.edu/download [Accessed April 5, 2020]

[7] J. Pennington, R. Socher and C. D. Manning, "GloVe: Global Vectors for Word Representation," 2014. https://nlp.stanford.edu/projects/glove/ [Accessed April 9, 2020]