Telecom Business Review
15 (1) 2022, 11-16
http://publishingindia.com/tbr/

Sentimental Analysis using Product Review Data

Amit Kumar*, Sonia Setia, Arjun Singh***, Thomas Abraham****, Yashaswi Shakya*****

*Assistant Professor, Sharda University, Greater Noida, Uttar Pradesh, India.
Email: amit.kumar23@sharda.ac.in

**Assistant Professor, Sharda University, Uttar Pradesh, Greater Noida, India.
Email: sonia.setia@sharda.ac.in

***Student, Sharda University, Greater Noida, Uttar Pradesh, India. Email: 2480arjunsingh@gmail.com

****Sharda University, Greater Noida, Uttar Pradesh, India. Email: imabrahomas@gmail.com

*****Sharda University, Greater Noida, Uttar Pradesh, India.
Email: 2019005329.yashaswi@ug.sharda.ac.in

Abstract

Our work systematically analyze the sentiment of product reviews and evaluate the correlation with their corresponding ratings. Sentiment analysis identifies the positive or negative mood represented in a piece of literature. Consumers write reviews with precise ratings on e-commerce platforms such as Amazon. We've noticed that there are occasionally discrepancies between the review and the rating. We performed deep learning guided sentiment analysis to identify such mismatches from amazon product review data. We convert reviews to vectors using paragraph vector and use them to develop a neural network using a GRU or gated recurrent unit our perspective makes advantage of both the semantic link between review content and product information.

Keywords: Sentiment Analysis, RNN, SVM, GRU

Introduction

The task of finding and computationally categorizing an author's sentiment in a document is known as sentiment analysis. It has global business applications, ranging from anticipating economic events driven by emotion displayed in articles and websites to assessing customer contentment and displeasure based on feedback and online activities. It is also the foundation for other applications such as recommender systems. Most websites now feature a field where users may give product or service evaluations. Valuable data, such as client feedback on things, causes for negative reviews, ideas, and so on may be gathered by doing sentiment analysis on submitted reviews. Users can also provide a number value to the product or service they are evaluating (i.e. a rating). Ratings on Amazon.com vary from one to five stars, with one being the worst of the five being the finest. In rare circumstances, there may be a discrepancy between a customer's review and rating. Because individual ratings are utilized to create the average rating, identifying reviews with mismatched values is crucial. Traditionally, a bag-of-words technique was used for converting a text for review into fixed-length feature vectors, which were then taken to instruct a surface classifier like Naive Bayes or a support vector (Kumar & Abirami, 2015). Pouch did surprisingly well and was popular for a long period, although it has two major faults. It ignores the semantic link between words and loses word ordering. For example, especially given the concept that "worst" should be closer to "least likeable" linguistically than "Las Vegas," the phrases "poor," "worst," and "Las Vegas" are all practically indistinguishable. Given the premise that the packet only examines word order in a confined context, it was confronted with scarce information and increasing functionality (Le & Mikolov, 2014). Machine learning has recently shown potential in the field of sentiment classification. When it comes to character creation, they have been demonstrated to outperform the pouch technique. (Mikolov et al., 2013) suggested a technique for properly expressing words in vector space by anticipating the existing word based on the surroundings and adjacent words. They were integrating semantically relevant words in vector space with semantically relevant phrases transferred to neighboring spots using an unsupervised technique. (Mikolov et al., 2013) proposed word vector, a non-supervised technique

for effectively capturing word meanings. Word vectors are vector representations of words, and lexically comparable words are nearer in vector space. We can also convey that, related terms like "good" and "excellent" are closer together than words like "fast," which are far away. Surprisingly, the vector depiction of words also catches language forms. For instance, the vector 'vec' ("King") – 'vec' ("Queen") + 'vec' ("Woman") shall give a vector which may be similar to the vector presentation of the word "Man". (Mikolov et al., 2013a) (Socher et al., 2013). An iterative neural tensor network for semantic communication across an emotion tree bank was proposed. For binary sentiment categorization of the Stanford opinion bank, their suggested model created state-of-the-art results. (Vieira & Moura, 2017) produced a simple yet powerful convolutional-neural-network (CNN) For the goal of sentiment analysis, it performed very well across several datasets. This network's input layer is formed of concatenated word2/vec word representations, proceeded by a convolution operation with several filters, a max bundling layer, and lastly a soft-max layer (Johnson & Zhang, 2014). In lieu of utilising low-dimensional word vectors as input to the Convolutional-Neural-Network, a variant of the bag-of-words model was utilised to build feature vectors (CNN). A parallel CNN was used by them with more than two convolutional layers to learn various sorts of embeddings from a single hot input vector. To carry out the sentiment analysis on review of movies obtained from rottentomatoes.com, (Ouyang et al., 2015) employed a 7-layer convolutional neural network architecture. 3 convolutional layers, 3 max-pooling layers, and a fully-connected layer with soft-max activation made up their architecture. Following up with the research of (Le & Mikolov, 2014; Mikolov et al., 2013) developed Section vectors is an unseen strategy for learning fixed-length featured vector for variable-length text like paragraphs, phrases, or full documents. Graph vector show how to surpass bag-of-words models and other types of depiction of textual format in a heterogeneity of classification tasks, encompassing sentiment analysis, obtaining state-of-the-art results. Because paragraph vectors are easier to provide training and get better effectiveness in catching syntactic and semantic linkages in a text, our model trained a low-dimensional vector of review text using them. Previously, much academics were solely concerned with the syntactic and semantic links among review papers. There is a high interest nowadays for integrating the users and information of product with the feature vector to improve it. (Tang et al., 2015) developed a neural network architecture which maintains the semantics of textual content and user sentiment. A customer was depicted as continuous matrix, and their sentiment was classified using the product of the word matrix and the user matrix by a neural network (Tang et al., 2015). A neural network model was also utilised for incorporating both user as well as product information into a document that shows the vector representation. They initially used a convolutional neural network to transform a document's text to continuous vector representation. Every person and product were represented as continuous matrices, and the resulting combination of user-text and product-text was put into a CNN for sentiment analysis. There were many IMDB datasets, Yelp 2014, and Yelp 2013 were utilised for sentiment analysis with their model. With all three datasets, they were able to reach cutting-edge findings (Chen et al., 2016). To build review embeddings for the IMDB and Yelp datasets, researchers employed a 1-layer CNN. Their CNN analyzed reviews of various lengths and generated results. Three hundred-dimensional vectors. They padded shorter reviews with zero vectors to accommodate varied durations of reviews. Filters with widths of 3 and 5 performed 1-dimensional convolution on the word embeddings and created several feature maps. Using max-overtime pooling in the pooling layer, only meaningful features were recorded. Multiple filters' outputs were then combined to generate a 300-dimensional vector. To train the network over K-classes, the Softmax function was utilised as an activation function. Further, they utilised a Recurrent Neural Network (RNN) for grabbing knowledge of temporary information and collect both product as well as user information, reporting state of the art results on the Yelp and IMDB datasets. The study founded that those products which first obtains favorable feedback is more likely to earn positive feedback in the future. Motivated by the results of (Sharma & Lin, 2013) model, the model also learnt product as well as temporal relations of reviews, in addition to syntactic and semantic relationships. These, we feel, are essential features that can boost sentiment categorization accuracy greatly. Our research is founded on the hypothesis that a popular product will obtain more high ratings, whereas a less popular product will receive more negative ratings. Recurrent neural networks (RNN) were utilised to collect this data since they excel at expressing sequences.

Methodology

We used RNN with GRU to build low-dimensional vector representations of reviews to analyze the sentiment of Amazon.com reviews, paragraph vectors and embedded vectors were utilized. Using paragraph vectors, we turned Amazon.com product reviews into fixed-length feature vectors. These feature vectors were then sorted in temporal

order and classified by product. With GRU, all groups were utilized to instruct an RNN. Product embeddings are the vectors created in the RNN's penultimate layer. Important information such as product attributes and chronological relationships between reviews are captured by these embeddings. We then trained a support vector machine by concatenating product embeddings with fixed-length vectors created by paragraph vectors. We also created a user interface to address the mismatches in review ratings. A person may write an extremely favourable review yet give it One or Two stars, or a strongly critical review but give it four or five stars in some cases (Sharma & Lin, 2013). Despite the fact that such instances are uncommon, they cause chaos in the head of people who are going to go through these evaluations. For solving this, we categorized reviews with a rating of One/Two as 'negative,' reviews whose rating was Three as 'neutral,' and reviews whose rating was Four/Five as 'positive.' After this, we made a web service application which predicts a class using our classifier. If anticipated class and those classes in which the ratings, which are submitted belongs disagree, the user is notified and given the opportunity to examine and change their rating. Our whole strategy is depicted in Fig. 1.

Pre-Processing Data

This study of information relied on a database of around three million five hundred thousand product reviews gathered from Amazon.com by the researchers (amazon review full dataset). Ratings, product ids, helpfulness, reviewer ids, review time, review title, and review texts are all included in each review. On a scale of One to Five, the rating is given. An overview of the dataset may be seen in Fig. 2. There are 2,200,000 Five-star rating, 622,308 Four-star rating, 265,684 Three-star rating, 171,153 Two-star rating, and 288,789 One-star rating, as seen in Fig. 3. Many terms in customer reviews, such as links, tags, informal language, and so on, have no major influence on the tone of reviews. Including such terms in reviews content will enhance the problem's dimensionalities. Before converting it to its associated vector the reviews are preprocessed to alleviate this issue. Our method involved the following steps:

- Links should be eliminated.
- Erasing whitespace between words.
- Converting casual terms like 'I'll' and 'I've' to their formal forms, such as 'I will' and 'I have,' and so on.

Punctuations are considered as different token to try to improve classifier accuracy.

Review Modeling using Paragraph Vectors

The structure of word vectors is heavily influenced by paragraph vectors (PV) (Le & Mikolov, 2014); (Mikolov et al., 2013). By anticipating the next word based on terms in a given context, the Word Vectors framework develops semantic linkages. Similarly, the PV framework get to know vectors by predicting the coming word in a paragraph from a huge number of situations. Every sentence corresponds to column of matrix D, each word corresponds to column of a matrix W. The model predicts the following word xi+3 given a context that has been taken from a paragraph or any document (for example, xi, xi+1, and xi+2). It accomplishes this by concatenating or merging the paragraph vector with word vectors in the situation. During training, the model is responsible to update the paragraph as well as word matrices to decrease error. The divided bag of letters architecture of sentence vector is shown in Fig. 5 (Le & Mikolov, 2014). This differs from PV-DM in that it disregards word order. PV-DBOW takes any context from any of the paragraphs, extracts any random word from the plethora of words from that context, and thereafter tries to forecast the context based on that word. It does not employ a word matrix, so there's less data to keep track of. PV-DM has been demonstrated to be more efficient than PV-DBOW in studies, thus we used PV-DM to make the review in our dataset to the fixed length vectors.

Studying Product Embedding using RNN+GRU

We turned 3.5 million product reviews into Three Hundred dimensional fixed-length vectors using paragraph vectors. We sorted reviews by 'product id' and then ranked them by'review time' to calculate product embeddings. Table 1 displays the sequence in which a review is organised for computing product embedding. The paragraph vector is used to replace the review text. The input sequence is temporally sorted vectors, and the targeted output is corresponding ratings, they are then utilised to instruct a gated recurrent unit (GRU) to understand embeddings for that particular product.

Recurrent Neural Networks (RNN)

In standard Neural Networks, inputs are treated as distinct entities. On the other hand, this method is useless for tasks like guessing the next word in a phrase. The prior word, or sequence of preceding words, decides the following word in this scenario. Because RNNs are adept at acquiring

sequential data, they are valuable in certain situations. RNNs' architecture includes loops that allow them to transfer data from previous inputs while analysing new data. The unrolling of a recurrent neural network over entire time step is seen in Fig. 6. Xt is input at step of time t, while Yt is output at the step of the same time in this architecture. The weight matrices U, V, and W must be learned, and St is the state that's hidden calculated at each time step t. Because it maintains information about input sequences based on prior inputs, St is also known as the network's memory. RNN computes a succession of hidden states and output for a given input sequence of x1, x2, x3,..., xT. Recurrent neural networks are trained via gradient descent of error, which is a back-propagation over time extension (BPTT).

Recurring Neural Network with Gated Feedback Unit

The gradient of errors with reference to each value matrix is equal towards update towards that matrix, and B-P-T-T uses the chain rule to compute the gradients. The value matrices U, V, and W are continuously upgraded through training algorithm when recurrent neural networks (R-N-N) are trained (B-P-T-T). As RNN is gaining relatively long context, the gradient begins to decrease (vanishing-gradient-problem) (VGP) or rising (VGP) and (exploding gradient problem) towards the initial stages depending on the activation functions, making training the network difficult. Using a gating technique, (GRU) were employed to overcome the vanishing gradient problem. The review vectors for each product are sorted by the time the review was filed, as shown in Table 1. The training sequence for that product was made up of these sorted review vectors and their related ratings. These sequences fed to a G-R-U were used to study a Product data is displayed as a 128-dimensional feature vector. Because the goal is to collect knowledge that already available.

Sentiment Classifications using SVM Methods

SVM or Support Vector Machine is a well-known deep learning technique for categorizing data that is both linear and nonlinear An SVM classifier searches for a hyper-plane as the classification model given a dataset of binary outcomes improves the dispersion between favorable and unfavorable samples. Using SVM, each review is labelled as "good," "neutral," or "negative." SVM adapts non-linear data to a higher dimension before discovering a linear hyper - plane to solve the problem. The Gaussian Radial Basis function is the kernel utilized for such nonlinear data (RBF).

Experimental Results and Evaluation

Sentiment Classification using Review Embedding

Using the Genism framework, we first created a sentence vector only model. Deeply fixed length vectors were created from 3.5 million Amazon.com product views. With a detection rate of 0.025, a ten-by-ten-inch screen, and 35 worker threads, we trained this model for 15 epochs. After each epoch, there was a decrease in learning rate by 0.002. A SVM classifier got learned using this vector and its accompanying rating (SVM). The performance of this SVM was assessed using the ten-fold cross-validation method. We calculated precision, recall, and classification accuracy to evaluate performance of our model. With only examine and evaluate, the model got an accuracy and recall score of 0.5861, recall value of 0.4087, and svm classifiers of 81.28 percent. These metrics are included in Table 2 for each progression as well as their average across ten iterations.

Table 1: Precision, Recall and Accuracy with Paragraph Vectors (w/o G-R-U)

Review. The embedding sequence for that specific product	1	0.5889	0.4051	81.25
Is the resulting value at the end of each product sequence's	2	0.5694	0.4072	81.15
last time step. We used a dropout rate of 0.25, Adam as a	3	0.5819	0.4092	81.28
stoc-hastic optimization approach, categorical cross entropy	4	0.5817	0.4086	81.26
as a loss function, a tem-porally distributeddense layer,	5	0.6003	0.4088	81.34
and 128 hidden units to train our GRU. During the training	6	0.6018	0.4077	81.37
stage, each unique prod-uct's product sequence was	7	0.5819	0.4097	81.39
obtained and saved in a file for subsequent retrieval.	8	0.5731	0.4081	81.20
	9	0.5859	0.4115	81.23
	10	0.5841	0.4114	81.45

Opinion Recognition with Review Embedding and Product Embedding

Then creating the three hundred-dimensional deep features for the ratings and reviews, then we sorted them by 'product ids' and grouped them by the 'review time,' with every group serving as insights to the (GRU). The embedding produced in last stage i.e. of the series was used as embedding of item's. This embedding of product was combined with the feedback embedding to instruct an (SVM). This dataset was validated using ten-fold cross validation. Just like with the sentence vector alone strategy, we calculated clarity, memory, and overall accuracy. These statistics are included in Table 3 for every iteration and also their mean across number of iterations. The scheme has an estimated precision of 0.5945, memory of 0.4252, and analysis efficiency of 81.82 percent using review embedding and product embedding. We improved detection precision from 81.29 to 81.82 percent, precision from 0.5860 to 0.5945, and memory from 0.408 to 0.4252 by utilizing both opinion and product embeddings. Fig. 1 compares the results of the two approaches.

Table 2: Preciseness, Recall and Accuracy with GRU + Paragraph Vector

1	0.5987	0.4183	81.69
2	0.5939	0.4317	81.90
3	0.5915	0.4265	81.86
4	0.5963	0.4253	81.89
5	0.5844	0.4310	81.79
6	0.5853	0.4231	81.94
7	0.5951	0.4252	81.81
8	0.6016	0.4301	81.84
9	0.5925	0.4206	81.83
Iteration	**Precision**	**Recall**	**Accuracy (%)**
10	0.6063	0.4412	81.77
Average	**0.5951**	**0.4250**	**81.80**

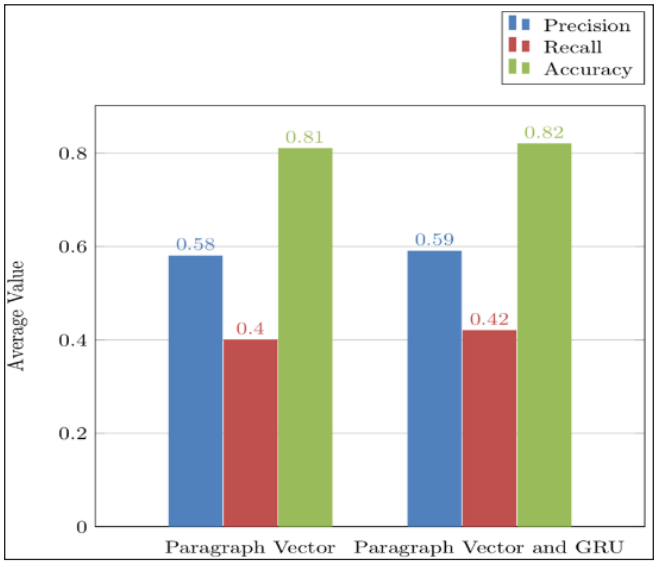

Fig. 1: Results on Model Trained with w/o GRU versus with GRU

Conclusion

In this article, we offer a machine training plan to address a significant challenge in e-commerce portals so when customer's opinion differs from the ranking. Before considering to acquiring particular item, ranking is a valuable reference for a potential client. We genuinely think that feedbacks with varying ratings can lead to confusing user experience, that's why we carried out this research project. Before creating the model, a paragraph vectors were appointed to understand the semantic and syntactic connection of 'review text.' Then we sorted, grouped, and united the review embeddings for producing product sequences, that would then be given to a GRU to grasp product embeddings. The SVM is instructed for sentiment categorization using a collection of review embeddings got from the sentence vector and item embeddings generated from the GRU. Our classifier operates at an accuracy of 81.29 with solo review embedding; nevertheless, incorporating em- bedding improves the accuracy to 81.82, that could

be used in sentiment analysis. to predict a review rating and contrast it to one already given If there is a difference among the stated ranking and the review, this webservice takes the 'review text' as well as the 'review score' and notifies to the reviewers.' In the synopsis, a demonstration on how the review item and word information can be paired for generating vigorous machine training model for opinion analysis. Similar approach, can be assumed, be used to learn user's information. If the results are compared to critical users, few individuals are lenient and might provide significantly greater scores for the same review text. This encourage us to use that same knowledge in the future.

References

Chen, T., Xu, R., He, Y., Xia, Y., & Wang, X. (2016). Learning user and product distributed representations using a sequence model for sentiment analysis. I*EEE Computational Intelligence Magazine, 11*(3), 34-44.

Johnson, R., & Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

Kumar, J. A., & Abirami, S. (2015). An experimental study of feature extraction techniques in opinion mining. *International Journal on Soft Computing, Artificial Intelligence and Applications (IJSCAI), 4*(1), 15-21.

Le, Q., & Mikolov, T. (2014, June). *Distributed representations of sentences and documents*. In International Conference on Machine Learning (pp. 1188-1196). PMLR.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems.*

Ouyang, X., Zhou, P., Li, C. H., & Liu, L. (2015, October). *Sentiment analysis using convolutional neural network*. In 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (pp. 2359-2364). IEEE.

Sharma, K., & Lin, K. I. (2013, April). Review spam detector with rating consistency check. In *Proceedings of the 51st ACM Southeast Conference* (pp. 1-6).

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013, October). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp. 1631-1642).

Tang, D., Qin, B., & Liu, T. (2015, July). Learning semantic representations of users and products for document level sentiment classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing* (volume 1: long papers, pp. 1014-1023).

Tang, D., Qin, B., Liu, T., & Yang, Y. (2015, June). *User modeling with neural network for review rating prediction*. In Twenty-Fourth International Joint Conference on Artificial Intelligence.

Vieira, J. P. A., & Moura, R. S. (2017, September). An analysis of convolutional neural networks for sentence classification. In *2017 XLIII Latin American Computer Conference (CLEI)* (pp. 1-5). IEEE.

Fang, X., & Zhan, J. (2015). Sentiment analysis using product review data. *Journal of Big Data, 2*(1), 1-14.

Bengio, Y., Ducharme, R., & Vincent, P. (2000). A neural probabilistic language model. *Advances in Neural Information Processing Systems, 13*.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval, 2*(1-2), 1-135.