Implementation and Comparison of Deep Learning, and Naïve Bayes for Language Processing

Abiodun Olalere

Department of Computer Science, ESPAM Formation University,

Cotonou, Benin

DOI: https://doi.org/10.51244/IJRSI.2024.1102044

Received: 20 February 2024; Accepted: 26 February 2024; Published: 27 March 2024

ABSTRACT

Text classification is one of the most important tasks in natural language processing, in this research, we carried out several experimental research on three (3) of the most popular Text classification NLP classifier in Convolutional Neural Network (CNN), Multinomial Naive Bayes (MNB), and Support Vector Machine (SVN). In the presence of enough training data, Deep Learning CNN work best in all parameters for evaluation with 77% accuracy, followed by SVM with accuracy of 76%, and multinomial Bayes with least performance of 69% accuracy. CNN has the best performance in the presence of large enough training dataset because of the presence of filter/ kernels which help to indentify patterns in text data regardless of their position in the sentence.

We repeated the training again with just one-third of our data, at this point SVM comes with the best performance, the performance of CNN noticeably drops but still better than Multinomial Naive Bayes, the reason why SVM works best when we reduce the training data was because of its ability to look for a hyper-plane that creates a boundary between different classes of data so as to properly classify them, so we believed that getting the hyper-plane was more efficient when we reduce the dataset, hence reason for the good performance. Multinomial Naive Bayes have the least performance which we attributed to its assumption of independency between the features which sometimes does not hold true.

We concluded that availability of data should be an important factor when choosing classifier for Natural Language Processing Text Classification task. CNN should be use in the presence of enough dataset, and SVM should be use when data is not enough. Multinomial Naive Bayes must not be trusted with state of the art NLP task due to its assumption of independency between the features.

Keywords: Text Classification, Natural Language Processing, Multinomial Naïve Bayes, Support Vector Machine, Deep Learning

INTRODUCTION

One of the most important problems in natural language processing is text classifications which are applicable in so many areas of modern technology and innovation. With text as one of the most common categories of data available, more than 80% of data are unstructured. This makes it difficult and challenging to comprehensively analyze them, hence many businesses are unable to exploit its full potential for business benefit [7]. The inability to exploit full potential of billions of unstructured data brings about the suitability of machine learning algorithm for training model for text classification and sentiment analysis in areas like

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XI Issue II February 2024

web search[4], spam detection[5],[23],[24] topic classification[6], news classification, sentiment classification[7] to the spotlight, as machine learning algorithms can be use in Natural Language Processing for text classification and sentiment[13,14,15,16] related activities such as spam filtering and other use of text classification as a core technique for machine learning process.

The aim of this research is evaluate and compare performances of each of the three of the most important classifier commonly used for state of the art natural language processing text classifier and sentiment analysis. We experimented this by implementing each of Linear Support Vector Machine, deep learning through Convolutional Neural Network (CNN), and multinomial variant of Bayesian classifier for text classification, after evaluation and comparison of the accuracy and performances of each classifier, we also dived into causes of variation and differences in their performances.

For all implementations in this research, we use stack overflow dataset which is publicly available at: https://storage.googleapis.com/tensorflow-workshop-examples/stack-overflow-data.csv dataset and can also be directly query from Google BigQuery platform.

RELATED WORK

1) Convolusional Neural Network (deep learning based)

Artificial neural network works to mimic functionality of human brain, input in artificial neural network represents the dendrites find in human brain while the different axion terminals then represent output from the neural network [15], [16], [22], [23]. In deep learning, a typical network contains one or more several hidden layers called deep neural network (Figure 1), and it works by performing computation on input data fed to the network to give an output.

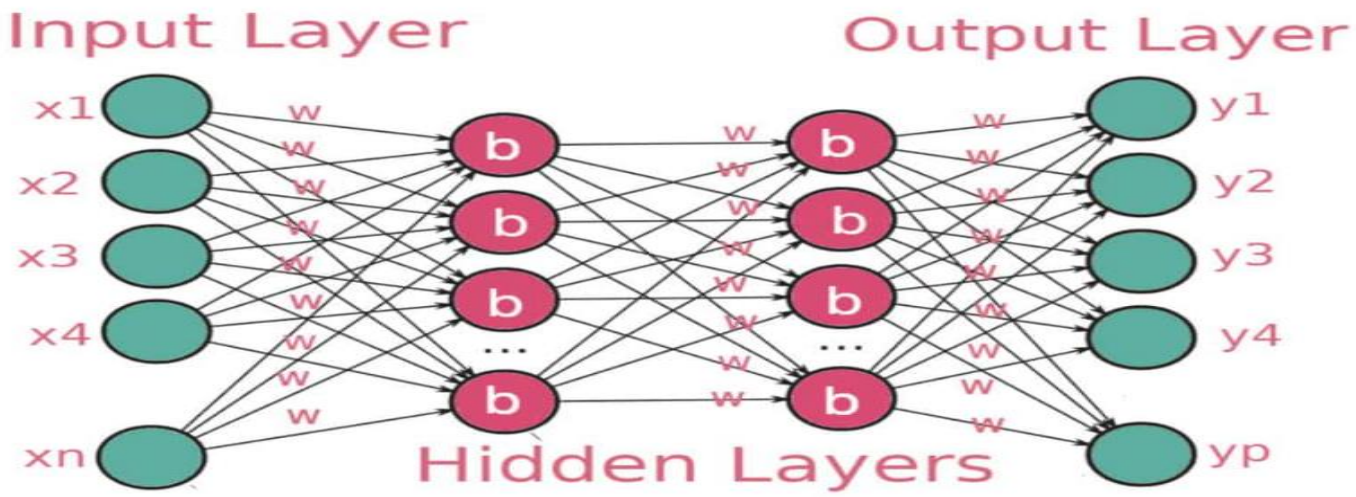

Figure 1 Hidden Layers of Convolutional Neural Network (CNN)

Convolution Neural Networks (CNNs) are complex multi-layered artificial neural networks with the abilityto detect complex features such as extraction of features in image and text in dataset. They are very efficient in computer vision task and so are commonly used for image segmentation, object detection, image classification, and text classification tasks [19, 20] for natural language processing.

Convolutional neural network has convolution layer and pooling layer (Figure 2) as the two major layers for separate purposes, and while the convolution layer obtains features from data, pooling layer reduces sizes of the feature map from the data.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XI Issue II February 2024

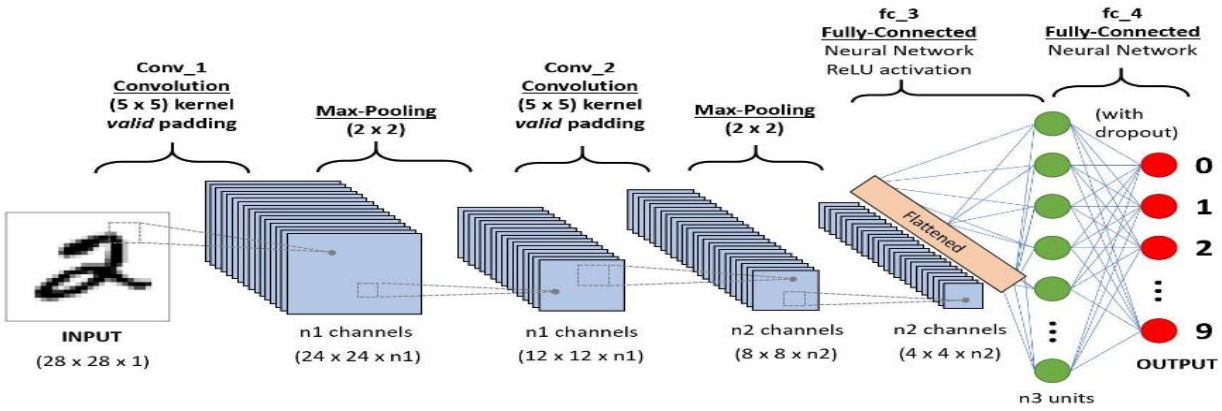

Fig 2 Connectivity of the Pooling and Kernels in CNN

During convolution, features are obtained from data, features that are obtained during convolution operation are fed to the CNN. Outputs from this convolution operation are the most important features and they are called feature map, a convolved feature, or an activation map. The output is computed by applying feature detector which is also known as kernel or filter to the input data. The next stage of the process is computation of the feature map by multiplication of the kernel and the matrix representation of the input data together, this process ensures that the feature map that is passed to the CNN is smaller but contains all essential features. This process is done by filter by going step by step process known as strides through every element in the input data.

2) Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine algorithm used for both classification and regression task [17], [18], [21]. It works by looking for a hyper-plane (Figure 3) that creates a boundary between two classes of data so as to properly classify them, it determines the best decision boundary between categories, hence they can be applied to vector which can encode data, and so text are classified into vector during text classification tasks by SVM algorithm.

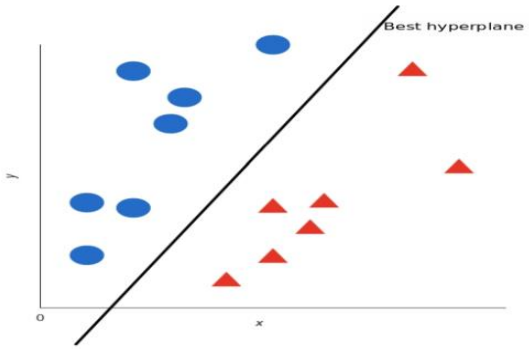

Figure 3 Hyper-plane mapping region of separation

Once the algorithm determined the decision boundary for each of the category in the dataset for which we want to analyze, we can proceed to obtain the representations of each of all the texts we want to classify in our NLP [9,10,11,12] and check for the side of the boundary that those representations fall into.

Theories supporting suitability of SVM for text classification

High dimensional input space: when training a classifier for text classification task, it is common to deal with several features, but the fact that SVM tries to prevent overfitting by using overfitting protective

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XI Issue II February 2024

measure which are independent on the number of features in the data gives SVM the potential to handle feature spaces in dataset, and hence their suitability for tasks relating to natural language processing.

Linearly separable properties of text categorization: Virtually, all of categories involved in text classification are linearly separable. And so, the fact that SVM works by finding linear separator between each of the categories in the textual dataset makes SVM suitable for tasks relating to NLP such as text classification, sentiment analysis, cyber troll and so on.

3) Multinomial Naïve Bayes (MNB)

Multinomial variant of Bayesian classifier is an algorithm commonly used for text classification related tasks and also problems with multiple classes [8]. In order to understand how Bayesian classifier works, it is important to understand the basic concept of Bayes theorem.

Tosin Ige, and Sikiru Adewale [9],[13],[14] successfully use multinomial Naïve Bayes algorithm to developed a machine learning based model along with an automated chatbot that can identify and intercept bully messages in an online chat conversation to protect potential victim with 92% accuracy. The accuracy of their model is a big discovery in Natural language processing field of artificial intelligence due to the mimicking of human intelligence to protect privacy and security across various online platform.

In Bayes theorem, the probability of an event occurring based on the prior knowledge of conditions related to an event is calculated based on the formula:

$P(A|B) = P(A) * P(B|A)/P(B)$

Where we are calculating the probability of class A when predictor B is already provided.

P(B) = prior probability of B

P(A) = prior probability of class A

P(B|A) = occurrence of predictor B given class A probability

Base on the principle of this calculation, we can automate the computation of tags in text and categorize them.

RESEARCH METHODOLOGY

1) Dataset: For each of the implementations in this research, we use stack Overflow questions and tags dataset which is publicly available at: https://storage.googleapis.com/tensorflow-workshop-examples/stack-overflow-data.csv and can also be query from Google BigQuery platform. The dataset contains two main column, the first column contains the question for all non-deleted Stack Overflow questions in the database while the second column is Tag which contains the tags on each of these questions.

2) Pre-processing and Feature Engineering: Since our dataset is a collection of several posts and tags from stackover flow dataset, it is imperative to clean the data of several unwanted characters. This necessitated the need for pre-procesing and feature engineering before we could actually use the data. As part of the pre-processing step, we removed unwanted characters from text, remove punctuation mark, stopwords, search for and decoding HTML, and so on, and then we finally split the dataset into training and validation dataset as part of our final data pre-processing steps.

After pre-processing followed feature engineering during which we converted each of the text to matrix of

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XI Issue II February 2024

token count by CountVectorizer. The count matrix is further converted to a normalized tf-idf representation also known as tf-idf transformer. Haven completed both pre-processing and feature engineering process, we then proceeded to training our model on SVM, Multinomial, and CNN classifiers.

3) Implementation

We use pipeline available which serves as a compound classifier in scikit-learn, each unique words was assigned an indices while we used tokenizer to count. Parameter for number of words was passed to the tokenizer so as to ensure that our vocabulary is limited to only top words. Haven done this, we were able to use our tokenizer along with texts_to_matrix method to create proper training data that could be passed to the model. After feeding our model with one hot encoded vector, and the transformation of features and labels to a format such that it could be read by keras deep learning python library. We trained the model by passing training data and labels, batch size and epochs to the fit () method. As for the SVM implementation, we useCountVectorizer() method, TfidfTransformer() method, after which we called SGDClassifier with the following configuration argument,

loss=’hinge’,

penalty=’l2',

alpha=1e-3,

random_state=42,

max_iter=5,

tol=None

RESULT AND DISCUSSION

In order to have perfect idea on the performances of each of classifier on Natural Language Processing text classification task, we used average precision score, average recall score, average F1-Score, and average accuracy as our evaluation parameters for comparison; we also observed some notable differences in training time for each of the classifier. We got different performances and result for each of the experimental implementation. Our first attention was drawn to the performance of Multinomial Naïve Bayes (Table 1) as it has the worst performance for each of the evaluation criteria among the three (3). In order to investigate this we, we used another dataset with few data to train the model, it was at this stage that we got a better result. This clearly proof to us that Multinomial variants of Bayesian classifier is good for small dataset, the reason for this is not far fetch, since multinomial naïve Bayes assume independency between features in the dataset, as the data gets bigger, the assumption of independency tends not to be true for some of the features in the dataset, hence the drop in performances with large dataset.

Table I: Metric Scores from Different Evaluation Criteria

Classifier	**Average Precision**	**Average Recall**	**Average F1-Score**	**Average Accuracy**
Multinomial Naïve Bayes (MNB)	0.72	0.69	0.68	0.69
Support Vector Machine (SVM)	0.77	0.77	0.76	0.76

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XI Issue II February 2024

Deep Learning (CNN)	0.78	0.76	0.77	0.77

As for Support Vector Machine (SVM) classifier whose performance is in-between that of Multinomial and CNN classifier (Figure 4). SVM works better than Multinomial for two reasons, the first reason is because it is not based on any independence assumption among the features, the second reason is because of its ability to look for a hyperplane that creates a boundary between two classes of data so as to properly classify them, as this enables it to determine the best decision boundary between the categories. Although, SVM works better than Multinomial Naïve Bayes for Natural Language Processing text classification tasks, it is also not suitable for a very large dataset due to complexity in training.

Figure 4 Comparing performances of CNN, SVM, and Multinomial across different evaluation metrics

For each of all the evaluation parameters for our experimental implementation, Convolutional Neural Network (CNN) has the best overall performance, with average precision (78%), average recall (76%), average F1-Score (76%), average accuracy (77%). We believed that deep learning convolutional neural network work best among the three (3) for natural language processing text classification task because it contains filter/ kernels which can help to identify patterns in text data, also the fact that these filters are translational invariant means that they can detect patterns regardless of their position in the sentence. Here, the convolutional architecture is able to identify -gram that could be use for prediction with the need to pre-specify any embedding vector for ngram.

CONCLUSION

In this research, we confirmed that of the three (3) of the most popular classifier for NLP text classification task, Convolutional Neural Network work best and even with all parameters for metric evaluation when we have enough training dataset, CNN is good for text classification in the presence of enough data because due to the presence of filter/ kernels which help to indentify patterns in text data regardless of their position in the sentence.

In the absence of enough training data, Support Vector Machine (SVM) works best which we believed to be due to its ability to look for a hyperplane that creates a boundary between different classes of data so as to properly classify them. Multinomial Naive Bayes works based on independent assumption between features which are sometimes not valid in real data, we believe this is the reason for its least performance among the three, and we believed that Multinomial Bayes classifiers must not be trusted for state of the Natural Language Processing (NLP) text classification task.

REFERENCES

1. Sentiment Analysis of Internet Posts," 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 2019, pp. 154-155, doi: 10.1109/ECICE47484.2019.8942736.
2. J. Zhao, L. Dong, J. Wu, K. Xu. Mood Lens: An Emoticon-Based Sentiment Analysis System for Chinese Tweets in Weibo. KDD 2012.).
3. Y. Chen and Z. Zhang, "Research on text sentiment analysis based on CNNs and SVM," 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), Wuhan, China, 2018, pp. 2731-2734, doi: 10.1109/ICIEA.2018.8398173.
4. Park D S, Chan W, Zhang Y, et al. Specaugment: A simple data augmentation method for automatic speech recognition[J]. arXiv preprint arXiv:1904.08779, 2019.
5. Faris H, Ala'M A Z, Heidari A A, et al. An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks[J]. Information Fusion, 2019, 48:67-83.
6. Watanabe K, Zhou Y. Theory-driven analysis of large corpora: Semisupervised topic classification of the UN speeches[J]. Social Science Computer Review, 2020: 0894439320907027.
7. Gao Z, Feng A, Song X, et al. Target-dependent sentiment classification with BERT[J]. IEEE Access, 2019, 7: 154290-154299.
8. Ige, T., & Adewale, S. (2022a). Implementation of data mining on a secure cloud computing over a web API using supervised machine learning algorithm. International Journal of Advanced Computer Science and Applications, 13(5), 1–4. https://doi.org/10.14569/IJACSA.2022.0130501
9. Ige, T., & Adewale, S. (2022b). AI powered anti-cyber bullying system using machine learning algorithm of multinomial naïve Bayes and optimized linear support vector machine. International Journal of Advanced Computer Science and Applications, 13(5), 5–9. https://doi.org/10.14569/IJACSA.2022.0130502
10. Park D S, Chan W, Zhang Y, et al. Specaugment: A simple data augmentation method for automatic speech recognition[J]. arXiv preprint arXiv:1904.08779, 2019.
11. Faris H, Ala'M A Z, Heidari A A, et al. An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks[J]. Information Fusion, 2019, 48: 67-83.
12. Watanabe K, Zhou Y. Theory-driven analysis of large corpora: Semisupervised topic classification of the UN speeches[J]. Social Science Computer Review, 2020: 0894439320907027.
13. Ige, Tosin, William Marfo, Justin Tonkinson, Sikiru Adewale, and Bolanle Hafiz Matti. "Adversarial Sampling for Fairness Testing in Deep Neural Network." arXiv preprint arXiv:2303.02874 (2023).
14. Okomayin, Amos, Tosin Ige, and Abosede Kolade. "Data Mining in the Context of Legality, Privacy, and Ethics." (2023).
15. Ige, Tosin, and Christopher Kiekintveld. "Performance Comparison and Implementation of Bayesian Variants for Network Intrusion Detection." 2023 IEEE International Conference on Artificial Intelligence, Blockchain, and Internet of Things (AIBThings). IEEE, 2023.
16. Adewale, Sikiru, Tosin Ige, and Bolanle Hafiz Matti. "Encoder-Decoder Based Long Short-Term Memory (LSTM) Model for Video Captioning." arXiv preprint arXiv:2401.02052 (2023).
17. Gao Z, Feng A, Song X, et al. Target-dependent sentiment classification with BERT[J]. IEEE Access, 2019, 7: 154290-154299.
18. Yong Z, Youwen L, Shixiong X. An improved KNN text classification algorithm based on clustering[J]. Journal of computers, 2009, 4(3): 230-237.
19. Sang-Bum Kim, Kyoung-Soo Han, Hae-Chang Rim and Sung Hyon Myaeng, "Some Effective Techniques for Naive Bayes Text Classification," in IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 11, pp. 1457-1466, Nov. 2006, doi:10.1109/TKDE.2006.180.
20. Nigam K, Lafferty J, McCallum A. Using maximum entropy for text classification[C]//IJCAI-99 workshop on machine learning for information filtering. 1999, 1(1): 61-67.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XI Issue II February 2024

21. Wang Z Q, Sun X, Zhang D X, et al. An optimal SVM-based text classification algorithm[C]//2006 International Conference on Machine Learning and Cybernetics. IEEE, 2006: 1378-1381.
22. Berger M J. Large scale multi-label text classification with semantic word vectors[J]. Technical report, Stanford University, 2015.
23. Wang S, Huang M, Deng Z. Densely connected CNN with multi-scale feature attention for text classification[C]//IJCAI. 2018: 4468-4474.
24. Guo B, Zhang C, Liu J, et al. Improving text classification with weighted word embeddings via a multi-channel TextCNN model[J]. Neuro computing, 2019, 363: 366-374.
25. Tao P, Sun Z, Sun Z. An improved intrusion detection algorithm based on GA and SVM[J]. Ieee Access, 2018, 6: 13624-13631.