Text Classification Using Machine Learning Techniques

K. Santoshi[1], U.Archana[1], D.Priyanka[2]

[1]Assistant Professor, Assistant Professor, Assistant Professor

Department of IT, GMRIT, AP

[2]Assistant Professor, Assistant Professor, Assistant Professor

Department of IT, AITAM, AP

ABSTRACT

Text classification is one of the most common way of sorting text into coordinated groups. It is otherwise called text tagging or categorization.Text classifiers can consequently examine text by utilizing Natural Language Processing (NLP), and afterward relegate a bunch of pre-characterized labels or classifications considering the substance. It helps users to get the required information for their queries within less period and the obtained text or data would be relevant to their search. Many methods and algorithms are used for the classification of text, but the accuracy varies from method to method. CNN can be used for many classification tasks in NLP. Convolutional Neural Networks have one or more convolutional layers. These CNN layers are defined as kernels. The matrix of kernel with fixed sizes moves over the input data. Getting your data in the right dimensions is extremely important for any algorithm. CNN increases the overall classification performance. In addition to that, the performance of each class is higher than 94%. This result indicates that CNN can be used for defense systems to meet high precision requirements.

I INTRODUCTION

Language is one of the strategies for correspondence with the assistance of which we can talk, read and compose. For instance, we think, we simply decide, plans, and more in normal language; exactly, in words. Nonetheless, the central issue that defies us in this era of Artificial Intelligence is that could we at any point discuss in much the same way with PCs. As such, might people at any point speak with PCs in their normal language?It is really difficult for us to foster NLP applications since PCs need organized or structured information, yet human discourse is unstructured and frequently questionable. Normal Language Processing (NLP) is a part of Artificial Intelligence (AI) that empowers machines to grasp the human language. It will probably fabricate systems that can figure out text and naturally perform undertakings like interpretation, spell check, or topic classification.

Actually, the primary assignment of NLP is program PCs for examining and handling colossal measures of regular language information. Language is intended for conveying about the world. By concentrating on language, we can come to see more about the world. We can take advantage of information about the world, in blend with phonetic realities, to construct computational regular language frameworks.

It is helpful to separate the whole language handling issue into two discussions:

- Lexical, semantic and syntactic information is utilized to handle the composed text of the language as well as the necessary real world data.
- All the data required above in addition to extra information about phonology is utilized to process spoke language as well as sufficient added data to deal with the further ambiguities that emerge in speech.

Text Classification suggests examining the fundamental feelings of a given text victimization language process (NLP) and elective procedures to separate a significant example of information and choices from a given gigantic corpus of text. It examines the sentiment and point of the creator towards the topic of the subject referenced inside the text. This text is a neighborhood of any record, posted via web-based entertainment or from any data supply. Text is named unbiased or abstract, opposite or correlative, or adjusted.

II.RELATED WORK

This study proposes an approach of automated classification of articles submitted via an online submission system. There are many text classification algorithms. In this study, the algorithms used are kNN, Naïve Bayes, and SVM algorithms, of which classification performances are evaluated effectively.

Text classification: In the context of the rapid development of digital information, text mining techniques play an important role in information and knowledge management, attracting the attention of researchers. Automated text classification (or categorization) is the division of an input textual dataset into two or more categories, in which each text can belong to one or more categories. Text classification is carried out to assign a predefined label or class to a text. For instance, a new article posted in an online newspaper can be assigned one of the categories such as technology, sports, or entertainment; each article published in a journal can be automatically classified into the topics of information technology, environment, fisheries, etc

Text classification algorithms:

There are many text classification algorithms. In this study, the algorithms used are kNN, Naïve Bayes, and SVM algorithms, of which classification performances are evaluated effectively.Related studies on text classification: Many studies on text classification have been applied to solve problems in practice. For example, applied SVM and decision tree to solve text classification problems, and compared their effectiveness with that of the classical decision tree algorithm. In addition, the singular value decomposition technique was applied to the SVM algorithm to shorten the dimension of characteristic space and reduce noise, making the classification process more effective. With the dataset of 7,842 texts on 10 various topics, 500 texts of each topic were randomly chosen to train, the remaining texts were to verify independence. The results showed that the classification by SVM is better than that by the decision tree. Moreover, using singular value decomposition to analyze and shorten the dimension of characteristic space helped improve the effectiveness of the SVM classifier. The classification results with accuracy were greater than 90%. The second effective algorithm was Naïve Bayes with an accuracy was 87.6%. However, the kNN algorithm classifier implementation gave low accuracy with 46.7%. Normally, when compared with maximum number of features, kNNalgorithm yieldsbetter results with minimum number of features.

III.METHODOLOGY

Classification using Natural Language Processing and Machine Learning:

Steps of article classification:

The automated classification of articles is divided into two phases. In the training phase, relying on the collected dataset with machine learning algorithms, the classification model generated is described in Figure 1.

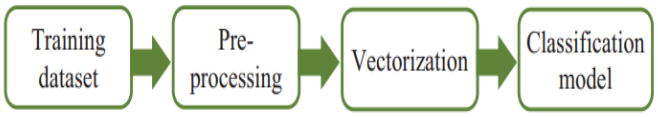


Figure 1: Training phase

In a testing phase, based on the classification model generated at the training step, articles are classified in the testing dataset. This stage is described in Figure 2

Figure 2: Testing phase

Data pre-processing:

File format conversion and word standardization:

Because the dataset used is .doc(x) files, it is necessary to convert them to plain text (.txt) for easy use in most algorithms and libraries serving automated classification. Converting format of an input article is based on Apache POI. Accordingly, Apache POI is used to perform read operations on the .doc(x) file, then write the readable content to the .txt file. After converting file format, word standardization is proceeded to convert all text characters into lowercase and delete spaces.

Word segmentation:

In Vietnamese, space does not segment words but separate syllables. Therefore, the segmentation phase is quite important in NLP. Currently, many tools have been successfully developed to segment Vietnamese words with relatively high accuracy. In this study, the VnTokenizer segmentation tool by was used. The tool was developed based on the integrated methods of maximum matching, weighted finite-state transducer, and regular expression parsing, using the dataset of Vietnamese syllabary and Vietnamese vocabulary dictionary. This automated Vietnamese segmentation tool segments Vietnamese text into vocabulary units (words, names, numbers, dates, and other regular expressions) with > 95% accuracy. The process of word segmentation using VnTokenizer is described in Figure 3.

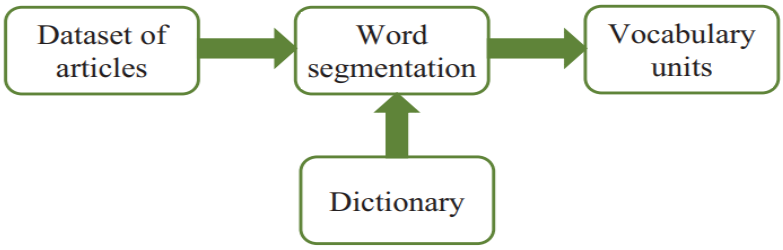


Figure 3:Process of word segmentation using VnTokenizer

Text vectorization:

There are several text representation models, i.e. vector space model based on the frequency weighting method, bag of words model, and graph-based model. In this study, the vector space model was applied. The vector space model can represent an unformatted text document as simple and formulaic notation. Because of its advantage, lots of research on the vector space model are being actively carried out. According to this model, each article is represented as a vector; each component of the vector is a separate term and is assigned a value that is the weight of this separate term.

CNN Architecture and methodology

Convolutional Neural Network(CNN) is a class of DNN and feed-forward artificial neural networks (where associations between hubs (nodes) don't frame a cycle). uses a variety of multi-facet perceptrons intended to require negligible preprocessing. These are roused by the animal visual cortex. It is an accurate method and performs so well in the classifications and Neural Networks. Here we are using Convolutional Neural Networks algorithm for Classification of text. Detecting multiple textures, edges, and corners are some of the features of convolutional layers. With this, all elements of a framework make CNN more sustainable to information of network structure. As the feature grows, we want to extend the element of the convolutional layer. Here we consider consecutive information of text data as data in time series and represented in one-dimensional matrix. To deal with one dimensional text data, we require a word embedding layer and an one-dimensional convolutional network.

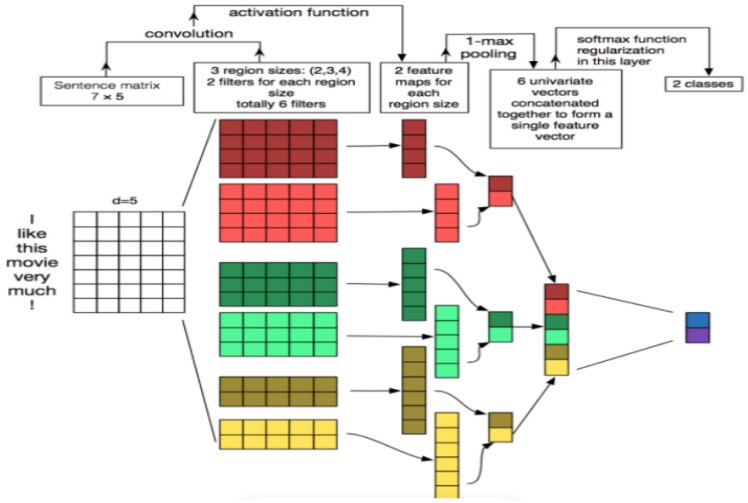


Figure 4. Convolutional Neural Network Layers

IV RESULT AND DISCUSSIONS

Text Classification is quite possibly of the Most fundamental pack in NLP categorizing text into organized groups. By utilizing Natural Language Processing (NLP), text classifiers can consequently anlyse text and afterward assign a set of pre-characterized labels or classifications in view of its content. By using CNN we are classifying tweets in a classified manner whether it is Positive, Negative, or Neutral. With help of the TensorFlow Library, the tweets have been classified. Firstly, the data have been cleaned up removing Emojis and Special characters, and this data is Pre-Processed by taking a dataset this dataset will be Split into the train(80%)and test(20%). Afterall, it rejects the unknown characters or strings into a tokenizer and fits the text data, and converts it into a

sequence. These inputs are converted into embedding layers and these are inserted into 1D convolution and the training and testing accuracy have been shown. The total number of positive is 6985 with an accuracy of 0.90 and the total negative tweets is 6549 with an accuracy of 0.90 and negative-positive reviews are 941 and positive-negative reviews are 525 and the weighted avg of 0.90.

```
# take a peek at the data
dataset.head()
```

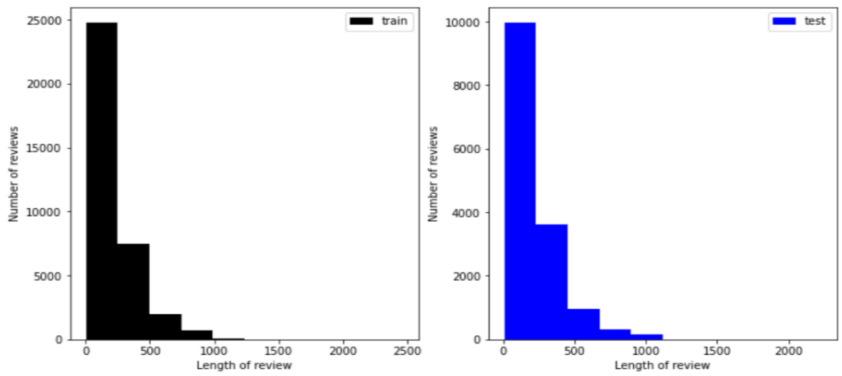
	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

Figure 5. Dataset took and modeled reviews

Graphical representation of the length of each review from training and testing datasets

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

train_lens = [len(s) for s in train_sequences]
test_lens = [len(s) for s in test_sequences]

fig, ax = plt.subplots(1,2, figsize=(12, 6))
h1 = ax[0].hist(train_lens,color="black")
ax[0].set_xlabel('Length of review')
ax[0].set_ylabel('Number of reviews')
ax[0].legend(['train'], loc='upper right')


h2 = ax[1].hist(test_lens,color="blue")
ax[1].set_xlabel('Length of review')
ax[1].set_ylabel('Number of reviews')

ax[1].legend(['test'], loc='upper right')
```

```
<matplotlib.legend.Legend at 0x7f16cdb56e50>
```



Creating the model with the below code and followed by the output obtained.

```
# create the model
model = Sequential()
model.add(Embedding(VOCAB_SIZE, EMBED_SIZE, input_length=MAX_SEQUENCE_LENGTH))
model.add(Conv1D(filters=128, kernel_size=4, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=4, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=32, kernel_size=4, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 1000, 300)         52738800

 conv1d (Conv1D)             (None, 1000, 128)         153728

 max_pooling1d (MaxPooling1D  (None, 500, 128)         0
 )

 conv1d_1 (Conv1D)           (None, 500, 64)           32832

 max_pooling1d_1 (MaxPooling  (None, 250, 64)          0
 1D)

 conv1d_2 (Conv1D)           (None, 250, 32)           8224

 max_pooling1d_2 (MaxPooling  (None, 125, 32)          0
 1D)

 flatten (Flatten)           (None, 4000)              0

 dense (Dense)               (None, 256)               1024256

 dense_1 (Dense)             (None, 1)                 257

=================================================================
Total params: 53,958,097
Trainable params: 53,958,097
Non-trainable params: 0
```

Training the model and evaluation of the created model following are the execution and outputs.

Model Training

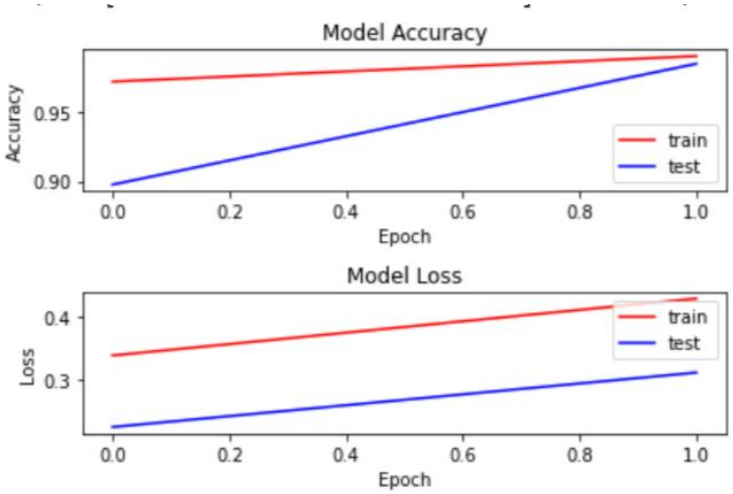
```
# Fit the model
model.fit(X_train, y_train,
          validation_split=0.1,
          epochs=EPOCHS,
          batch_size=BATCH_SIZE,
          verbose=1)
```

```
Epoch 1/2
247/247 [==============================] - 60s 190ms/step - loss: 0.4038 - accuracy: 0.7844 - val_loss: 0.2679 - val_accuracy: 0.8874
Epoch 2/2
247/247 [==============================] - 46s 187ms/step - loss: 0.1272 - accuracy: 0.9548 - val_loss: 0.2680 - val_accuracy: 0.8977
<keras.callbacks.History at 0x7f16cda87d90>
```

Model Evaluation

```
[22] predictions = (model.predict(X_test) > 0.5).astype("int32")
     predictions[:10]

     array([[0],
            [1],
            [0],
            [1],
            [1],
```

```
[24] from keras.callbacks import ModelCheckpoint, EarlyStopping
     from keras.callbacks import EarlyStopping
     early_stopping = EarlyStopping(monitor='val_loss', patience=2)
     plt.subplot(2,1,1)
     history1 = model.fit(X_train, y_train, validation_split=0.1, epochs=2, callbacks= [early_stopping], verbose=1)
     history2 = model.fit(X_test, y_test, validation_split=0.1, epochs=2, callbacks= [early_stopping], verbose=1)
     plt.plot(history1.history['accuracy'], 'r', history2.history['accuracy'], 'b')
     plt.title('Model Accuracy')
     plt.ylabel('Accuracy')
     plt.xlabel('Epoch')
     plt.legend(['train', 'test'], loc='lower right')


     plt.subplot(2,1,2)
     plt.plot(history1.history['val_loss'], 'r', history2.history['val_loss'], 'b')

     plt.title('Model Loss')
     plt.ylabel('Loss')
     plt.xlabel('Epoch')
     plt.legend(['train', 'test'], loc='upper right')

     plt.tight_layout()
```



Finally, the classification report is as follows.

```
[25] # Final evaluation of the model
     scores = model.evaluate(X_test, y_test, verbose=1)
     accuracy=print("Accuracy: %.2f%%" % (scores[1]*100))


469/469 [==============================] - 7s 16ms/step - loss: 0.0361 - accuracy: 0.9895
Accuracy: 98.95%
```

```
from sklearn.metrics import confusion_matrix, classification_report

labels = ['negative', 'positive']
print(classification_report(test_sentiments, predictions))
pd.DataFrame(confusion_matrix(test_sentiments, predictions), index=labels, columns=labels)
```

```
              precision    recall  f1-score   support

    negative       0.93      0.87      0.90      7490
    positive       0.88      0.93      0.91      7510

    accuracy                           0.90     15000
   macro avg       0.90      0.90      0.90     15000
weighted avg       0.90      0.90      0.90     15000
```

	negative	positive
negative	6549	941
positive	525	6985

V CONCLUSION

In view of natural language processing and deep learning algorithms, this study proposed a solution for text classification of articles/reviews to help authors/editors save time and effort when processing articles/reviews on the system. Data pre-processing steps are significant to making classification datasets in a standardized format for running the three algorithms of SVM, Naïve Bayes, and CNN. The results showed that the CNN algorithm gives better classification performance than the remaining classifiers. The rate of accuracy has been very consistent. This CNN is very elegant in handling larger datasets. Experimental outcomes demonstrate the way that our model can work on the performances of a group of related errands by investigating normal features. In future work, we might want to examine the other sharing systems of the undertakings.

REFERENCES

[1] T.T. Dien, N.T Thanh-Hai, N. Thai-Nghe "Deep Learning Approach for Automatic Topic Classification in an Online Submission System," Advances in Science, Technology and Engineering Systems Journal, vol. 5, no. 4, pp. 700-709 (2020).

[2] M. Thangaraj, M. Sivakami, "Text Classification Techniques: A Literature Review," Informing Science Institute, Interdisciplinary Journal of Information, Knowledge, and Management, Volume 13, 2018

[3] JasleenKaur, Dr.Jatinderkumar R. SAINI, "A Study of Text Classification Natural Language Processing Algorithms for Indian Languages," VNSGU Journal Of Science And Technology, Vol.4, No.1, July 2015 162 - 167, ISSN:0975-5446.

[4] DuyDuc An Bui, Qing Zeng-Treitler, "Learning regular expressions for clinical text classification," Journal of the American Medical Informatics Association, Volume 21, Issue 5, September 2014, Pages 850–857, https://doi.org/10.1136/amiajnl-2013-002411

[5] Vijay Garla, Caroline Taylor, Cynthia Brandt, "Semi-supervised clinical text classification with Laplacian SVMs: An application to cancer case management," Journal of Biomedical Informatics, Volume 46, Issue 5, 2013, Pages 869-875, ISSN 1532-0464, https://doi.org/10.1016/j.jbi.2013.06.014.

[6] T. T. Dien, B. H. Loc and N. Thai-Nghe, "Article Classification using Natural Language Processing and Machine Learning," 2019 International Conference on Advanced Computing and Applications (ACOMP), 2019, pp. 78-84, DOI: 10.1109/ACOMP.2019.00019.

[7] DuyDuc An Bui, Guilherme Del Fiol, Siddhartha Jonnalagadda, "Text Classification to leverage information Extraction from Publication reports," Journal of Biomedical Informatics, Volume 61, 2016, Pages 141-148, ISSN 1532-0464, https://doi.org/10.1016/j.jbi.2016.03.026

[8] Romanov, A, et al. 2019. "Application of Natural Language Processing Algorithms to the Task of Automatic Classification of Russian Scientific Texts," Data Science Journal, 18: 37, pp. 1–17. DOI: HTTPS:// doi.org/10.5334/dsj-2019-037

[9] XiaoyuLuo, "Efficient English text classification using selected Machine Learning Techniques," Alexandria Engineering Journal, Volume 60, Issue 3, 2021, Pages 3401-3409, ISSN 1110-0168, https://doi.org/10.1016/j.aej.2021.02.009.

[10] Liu, P., Qiu, X., & Huang, X. (2016). "Recurrent neural network for text classification with multi-task learning." arXiv preprint arXiv:1605.05101.