BBC News Text Classification

1st Kam Say Hong
School of Computer Science
University Science Malaysia
Penang, Malaysia
kamsayhong@student.usm.my

2nd Lee Chun Yip
School of Computer Science
University Science Malaysia
Penang, Malaysia
l_chunyip@student.usm.my

***Abstract*—Text classification is widely studied and implemented in real application of various domains. One of the common application is text classification for news articles in media and communication domains, which helps to foster the process of classifying the news articles into the right categories. In this study, several classifiers are developed and compared, in order to determine the best-performing classifier for the text classification of BBC news articles. Furthermore, the effect of text pre-processing on the classifier's performance is also within our scope of study. The data with different setting of text pre-processing steps is used to train and evaluate the classifiers, and the outcomes are used for comparison purposes. Moreover, the effect of N-gram order on the classifier's performance is studied, with evaluation of classifier's performance across the analysis of increasing N-gram order which are the unigram, bigram and trigram.**

Keywords—text classification, BBC news article, N-gram, text pre-processing.

I. INTRODUCTION (*HEADING 1*)

The news article is a communication medium that has been widely used to inform and educate readers on the latest issues. The issues are generally categorized into various domains such as business, technology, sport, entertainment, politics and etc.. The purpose of the categorization is to allow the readers to choose to read the articles with their favorable domains.

The outcomes of extracting insight from the unstructured textual data are fruitful, yet the process is time-consuming and tedious, and prone to human errors if the task is done manually. A more feasible approach for this task is the implementation of a machine learning model that has been trained for text classification purposes. Text classification is the process of categorizing texts into predefined groups [1], which is one of the important topics in natural language processing. There are binary classification and multi-class classification in the context of text classification. Binary classification categorizes the data into two groups, which is generally applied in sentiment analysis. Multi-class classification can be further divided into single-label multi-class classification and multi-label multi-class classification [2], which have more than two groups. Thus, the text classification task for news articles is a multi-class classification.

In this paper, several single-label multi-class classification classifiers are developed in order to classify news articles into the right category based on their text contents, and the best-performed classifier is selected based on the evaluation metrics.

II. PROBLEM STATEMENT

Generally, the study of text classification models for English language is common nowadays, as the research works for this particular language have been a while. The studies regarding text classification commonly focused on the comparison in terms of performance between the classifiers. To improve the performance classifier, text pre-processing must be done in order to ease the conversion of unstructured data to be a machine-readable format in the next phase. In a research work proposed by X. Luo in the year 2021 [3], multiple text classification classifiers are developed for classification tasks on English text and documents, which are SVM, Naive Bayes and Logistic Regression-based classifiers. Another research work proposed by F.Fanny, Y.Muliono, and F.Tanzil in year 2018 [4], the authors also had developed several text classification models which are SVM, Naïve Bayes and K-Nearest Neighbour (KNN) for the text classification tasks on news articles from multiple sources. However, the authors have studied the effect of different stemmer algorithms on the accuracy of the classifiers, and it showed that the stemmer algorithms did have an effect on the classifier's performance.

It is hard to find any studies which focused on the effect of the text pre-processing step on the classifier's performance. This phenomenon is also applied in the case of word embedding methods, especially N-Gram. A study on the effect of N-Gram sequence variation on the classifier's performance is an interesting topic that the researcher should focus on.

III. OBJECTIVES OF STUDY

As the development of classifiers for the classification tasks on news articles has widely been done throughout these years, this study will encompass more aspects on the performance of classifiers. To ensure the success of this study, a total of three objectives are formed in this study. The first objective is to explore and identify the best-performing classifier for the text classification tasks. The second objective is to study the effect of text pre-processing on the performance of the classifier. The third objective is to study the effect of N-Gram sequences on the performance of the classifier.

IV. DATASET DESCRIPTION

The dataset is retrieved from the Kaggle database, with the name BBC News Summary. The dataset contains a total of 2225 news articles in .txt format, from the BBC news website corresponding to five domain areas from the year 2004 to 2005, which are business, entertainment, politics, sport and technology. The number of news articles for each domain is shown in Table I. Based on the chart in Fig.1, the data for each domain is considered balanced.

TABLE I NUMBER OF NEWS ARTICLES FOR EACH DOMAIN

Domains	Number of News Articles
Business	510
Entertainment	386
Politics	417
Sport	511
Technology	401
Total	2,225

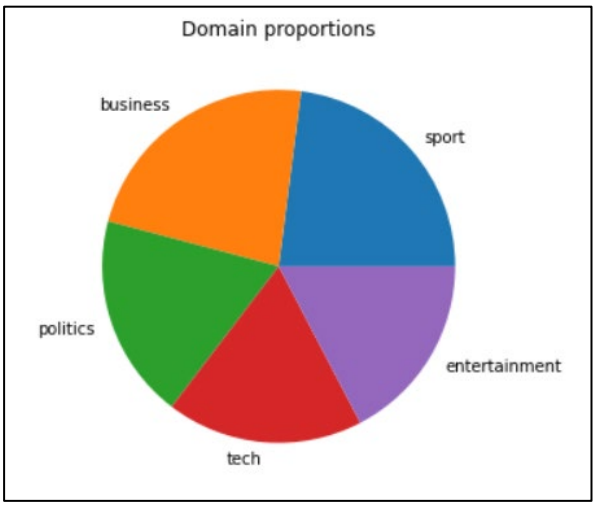

Fig. 1 Pie Chart of News Article Domains Proportion

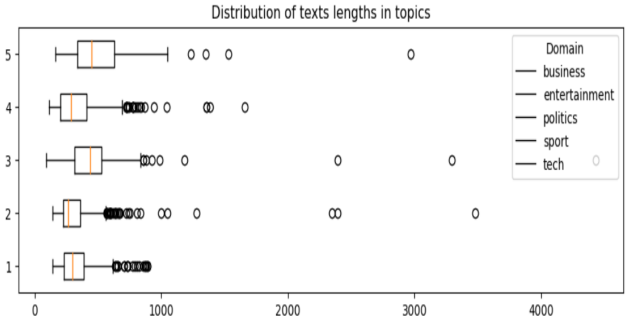

Fig. 2 Distribution of texts length in topics

Based on the boxplot in Fig.2 , the text lengths for articles in the business domain tend to be longer than in other domains. Besides, outliers are present in every domain. Generally, the distribution of text length for all domains is positively skewed, except the politics domain that showed a negatively skewed distribution. The positively skewed distribution indicated there is a high frequency of articles with long text length, and vice versa.

The frequency distribution of each category after stop words removal is shown in Table II as below:

TABLE II TOP 20 FREQUENT TEXT FOR EACH DOMAIN

Domain	Top 20 Frequent Text
Business	us, year, would, also, new, market, growth, company, last, economy, firm, could, bank, economic, sales, government, oil, 2004, years, may
Entertainment	Film, best, music, also, us, one, years, new, show, first, last, awards, year, number, award, uk, films, two, director, tv
Politics	Would, government, labour, people, election, blair, party, also, new, could, minister, told, brown, public, plans, howard, uk, one, prime, say
Sport	Game, first, would, win, last, world, england, one, two, also, time, back, players, play, cup, new, good, team, side, second
Tech	People, also, new, technology, would, could, one, mobile, games, users, music, use, digital, software, us, many, like, net, make, get

V. PROPOSED SOLUTION

Our proposed solution is to classify the news into their respective categories by using machine learning approaches. To realize this goal, we set up 3 experiments.

Experiment 1 is to identify the best performing classifier. Given that there are labelled data, we will use supervised machine learning algorithm to classify them. The result of those classifiers will then be recorded, compared and evaluated to identify the best performing classifier. The machine learning algorithms that are used for text classification in this project are shown in Table III.

TABLE III ALGORITHMS AND THEIR BRIEF DESCRIPTION

Machine Learning	Description
Naïve Bayes	An algorithm that works on Bayes theorem and have high independence assumption
Support Vector Machine (SVM)	Works by mapping data into high dimensional space to classify it
Logistic Regression	Using mathematical concept to find the relationship between datapoint

Experiment 2 is to identify the effect of n-gram sequence towards our solution. We will experiment with unigram, bigram and trigram using our classifiers and record the results. Those result will then be evaluated to identify their effect towards our models.

Experiment 3 will deal with the text pre-processing techniques where we will evaluate the effect of stop words removal, stemming and lemmatization towards our model performance. We will record result before and after text pre-processing techniques and compare them to get better understanding of their effect towards our models.

VI. METHODLOGY

A. Performance Metrics

The performance metrics used in this project are accuracy, precision, recall and f1 score. The determinant performance metrics in this project is f1 score.

The formula of the metrics is shown in Table IV as below:

TABLE IV FORMULA OF PERFORMANCE METRICS

Metrics	Formula
Accuracy	$\frac{TP + TN}{N}$
Precision	$\frac{TP}{TP + FP}$
Recall	$\frac{TP}{TP + FN}$
F1	$\frac{2 * Precision * Recall}{Precision + Recall}$

Where TP represents True Positive, TN represents True Negative, FP represents False Positive, FN represents False Negative and N represents total number of data.

B. *Data spliting*

The data is split using 80:20 ratio where 80% of data is used as training set while the rest is used as testing set

C. *Expermient 1*

The framework of experiment 1 is illustrated in Fig.3 as below:

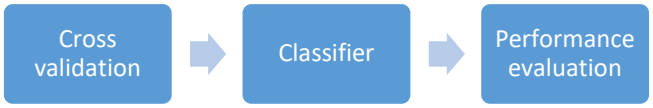

Fig. 3 Experiment 1 Frameworks

In this experiment, we use the built-in function of CountVectorizer in python to pre-process the data before we move into cross validation stage. The pre-processing steps include punctuation removal, stop words removal and tokenization. After that, we map it to the dtm to carry out the cross validation. The cross-validation technique used is 5-fold cross validation given that our data is small size in nature. The result of cross validation is recorded and used to fine tune our hyperparameter in next stage

A similar process is done in classifying stage. The only difference is we use the test set instead of training set in previous stage. Results including accuracy, precision, recall and f1 are recorded and compared among three classifiers: Naïve Bayes, SVM and Logistic Regression.

D. *Experiment 2*

In experiment 2, our focus is to analyse the effect text pre-processing to our model. The text pre-processing techniques chosen in this experiment are stop words removal, lemmatization and stemming. The Table V below briefly explain the techniques:

TABLE V TEXT PRE-PROCESSING STEPS INVOLVED

Text pre-processing	Description
Stop words removal	Removal of words that frequently appears and bring no meaning to the task
Lemmatization	Convert words into their root form
Stemming	Stripping the suffix from the end of words

We will test each technique, and records results before and after those techniques. The framework of experiment 3 is illustrated in Fig.4 below:

Fig. 4 Experiment 2 Framework

In this experiment, we will fit the text pre-processing techniques into each classifier. Then the result is recorded and evaluated.

E. *Experiment 3*

In experiment 3, our objective is to understand effect of n-grams to our models. The framework of our experiment is shown in Fig.5 as below:

Fig. 5 Experiment 3 Frameworks

The framework of experiment 3 is similar to experiment 2 except in this case we are varying n-gram from unigram to trigram. We train our data in the order of unigram, bigram and trigram then fit them into our test set and record their performance.

The text pre-processing done in this experiment are stop words removal, tokenization and punctuation removal. They remain constant in this experiment.

VII. ANALYSIS AND FINDINGS

A. *Experiment 1*

Table VI CLASSIFIERS PERFORMANCE SCORES (AFTER STOPWORD REMOVAL)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.9640	0.9643	0.9640	0.9639
SVM (kernel = Poly, c =1)	0.4809	0.7982	0.4809	0.4571
Naïve Bayes	0.9798	0.9800	0.9798	0.9798
Logistic Regression (Penalty = 'none')	0.9730	0.9735	0.9730	0.9730
Logistic Regression (Penalty = 'l2')	0.9730	0.9732	0.9730	0.9729

Based on the results in Table VI, the Naïve Bayes classifier has the highest scores for all performance metrics among the five classifiers. The F1-score is selected for the evaluation metric, as both proportions of false positive and false negative results are crucial in our text classification task. The Naïve Bayes classifier has the highest F1-score, so it is selected as the best-performing classifier.

B. *Experiment 2*

Table VIII CLASSIFIERS PERFORMANCE SCORES (BEFORE STOPWORD REMOVAL)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.9551	0.9557	0.9551	0.9551
SVM (kernel = Poly, c =1)	0.6404	0.7650	0.6404	0.6283
Naïve Bayes	0.9820	0.9823	0.9820	0.9820
Logistic Regression (Penalty = 'none')	0.9551	0.9550	0.9551	0.9548
Logistic Regression (Penalty = 'l2')	0.9573	0.9577	0.9573	0.9571

Table VIIII CLASSIFIERS PERFORMANCE SCORES (AFTER STOPWORD REMOVAL AND STEMMING)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.9618	0.9620	0.9618	0.9616
SVM (kernel = Poly, c =1)	0.4831	0.7973	0.4831	0.4571
Naïve Bayes	0.9820	0.9825	0.9820	0.9821
Logistic Regression (Penalty = 'none')	0.9708	0.9710	0.9708	0.9706
Logistic Regression (Penalty = 'l2')	0.9708	0.9709	0.9708	0.9706

Table IX CLASSIFIERS PERFORMANCE SCORES (AFTER STOPWORD REMOVAL AND LEMMATIZATION)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.9640	0.9641	0.9640	0.9639
SVM (kernel = Poly, c =1)	0.4831	0.7973	0.7973	0.4571
Naïve Bayes	0.9820	0.9825	0.9820	0.9821
Logistic Regression (Penalty = 'none')	0.9730	0.9734	0.9730	0.9730
Logistic Regression (Penalty = 'l2')	0.9708	0.9709	0.9708	0.9706

Table X DESCIPTION FOR TEXT PRE-PROCESSING STEPS INVOLVED IN EACH SETTING

Setting	Description
Textpre_1	Before stopwords removal (undergone punctuation removal and tokenization)
Textpre_2	After stopwords removal (results based on experiment 1, which undergone punctuation removaltokenizationation)
Textpre_3	After stopwords removal and stemming (undergone punctuation removal and tokenisation)
Textpre_4	After stopwords removal and lemmatization (undergone punctuation removal and tokenization)

Table VII, VIII and IX results show that the Naïve Bayes classifier has the highest scores for all performance metrics. The F1-score for each classifier in different pre-processing steps are shown in Fig.6 and Fig.7. For Naïve Bayes classifier, the F1-score dropped after the stopwords removal. However, the F1-score increases after the stopword removal and stemming, and the same goes for stopword removal and lemmatization.

In contrast, the F1-score of Logistic Regression (Penalty = 'l2') classifier increased after the stopword removal. However, the F1-score is slightly dropped after the stopword removal and stemming, and after the stopword removal and lemmatization.

F1-score of Logistic Regression (Penalty = 'none') classifier also showed a similar trend, yet the F1-score after the stopword removal and lemmatization same as the score after the stopword removal. The results indicated that the lemmatization has no improvement on the performance score for this classifier. A similar trend also observed in the F1-score of SVM with linear kernel.

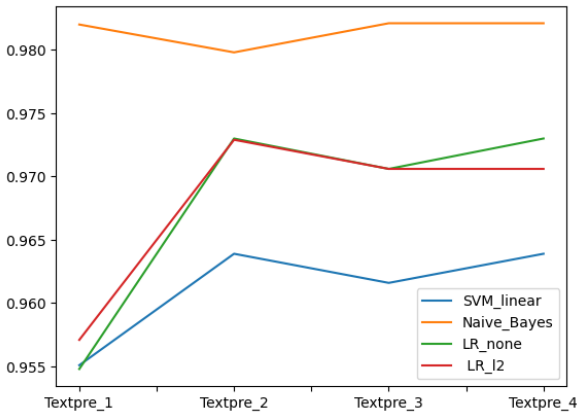

Fig. 6 F1-Score of classfier for Different Sets of Pre-Processing Steps Setting (SVM with linear kernel, Naïve Bayes, Logistic Regression)

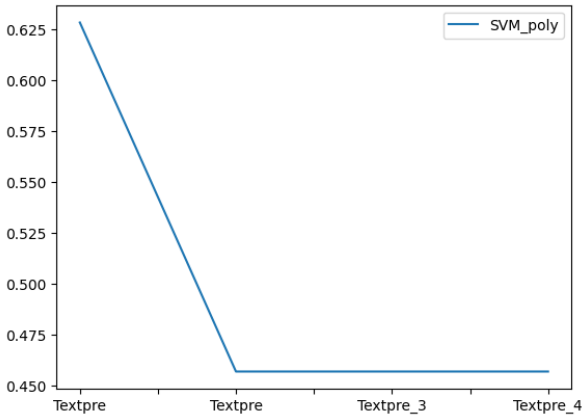

Fig. 7 F1-Score of classifier for Different Sets of Pre-Processing Steps Setting (SVM with polynomial kernel)

As for SVM with polynomial kernel classifier, the F1-score is the highest before the stopwords removal among all the pre-processing steps. Any pre-processing steps in this classifier would resulted in sharply decrease in the F1-score.

C. *Experiment 3*

Table XI CLASSIFIERS PERFORMANCE SCORES (UNIGRAM)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.9685	0.9686	0.9685	0.9684
SVM (kernel = Poly, c =1)	0.5146	0.8176	0.5146	0.4959
Naïve Bayes	0.9820	0.9823	0.9820	0.9820
Logistic Regression (Penalty = 'none')	0.9685	0.9688	0.9685	0.9684
Logistic Regression (Penalty = 'l2')	0.9730	0.9731	0.9730	0.9729

Table XII CLASSIFIERS PERFORMANCE SCORES (BIGRAM)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.9079	0.9123	0.9079	0.9072
SVM (kernel = Poly, c =1)	0.3663	0.7977	0.3663	0.3094
Naïve Bayes	0.9506	0.9509	0.9506	0.9505
Logistic Regression (Penalty = 'none')	0.9348	0.9360	0.9348	0.9345
Logistic Regression (Penalty = 'l2')	0.9326	0.9349	0.9326	0.9321

Table XIII CLASSIFIERS PERFORMANCE SCORES (TRIGRAM)

Classifier	Accuracy	Precision	Recall	F1
SVM (kernel = Linear, c =1)	0.6225	0.7721	0.6225	0.6404
SVM (kernel = Poly, c =1)	0.2921	0.8368	0.2921	0.2415
Naïve Bayes	0.7191	0.8119	0.7191	0.7322
Logistic Regression (Penalty = 'none')	0.7213	0.8180	0.7213	0.7354
Logistic Regression (Penalty = 'l2')	0.6787	0.8013	0.6787	0.6948

Table XIV OVERLAPPED WORDS BETWEEN DOMAIN CATEGORIES FOR EACH N-GRAM ORDER

N-Gram Order	Overlapped Words between Domain Categories
Unigram	'said', 'mr', 'music', 'new', 'year', 'people'
Bigram	'new york', 'told bbc', 'said mr'
Trigram	'told bbc news', 'told bbc radio'

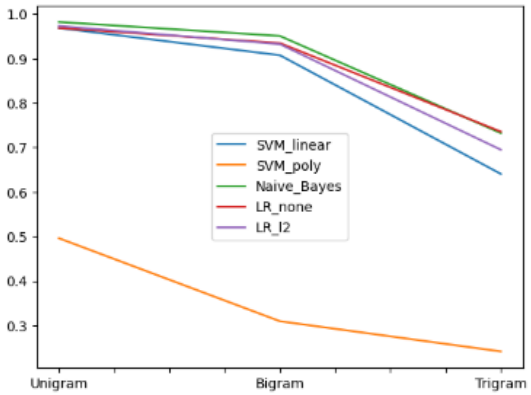

Fig. 8 F1-Score of classifier for Different Order of N-gram

Table XI and XII results show that the Naïve Bayes classifier has the highest scores for all performance metrics in both unigram and bigram analysis. Table XIII results show that the Logistic Regression (Penalty = 'none') classifier has the highest scores for all performance metrics in trigram analysis. Based on Fig.8, the F1-score for all classifiers shows a similar trend, whereby the score decreases across the N-Gram orders. Table XIV showed the overlapped words between domain categories for each N-Gram order. The overlapped words are identified based on the top 10 most frequently appearing words in the Word N-Gram analysis of each domain category.

VIII. DISCUSSION

For text classification tasks, the text data has to convert into the machine-understandable format, which is the numerical representation known as vectors, before feeding it to the machine learning classifier. The count vectorizer used in this study helps to convert the collection of text documents to a matrix of token counts. In other words, the data is converted to a list of words associated with the occurrence of each word in the document, which is also known as the bag of words. The bag of words only emphasizes the occurrence of each word, regardless of the word order.

Based on the outcomes in Experiment 1, the Naïve Bayes classifier is the best-performing model for the text classification task of BBC news articles. The Naïve Bayes classifier is a probabilistic model based on the Bayes theorem, with an assumption of independence among the attributes [5]. The Naïve Bayes Classifier is widely used for text and document classification due to its simplicity and efficiency. The classifier has to be trained with well-categorized text documents to identify and compare the text contents in all categories with the bag of words model. Subsequently, the bag of words representation is used for text classification on unseen text documents based on the posterior probability [6]. The Naïve Bayes classifier is able to handle a large number of attributes, which is the words in this study. The presence of irrelevant attributes or words would not affect the performance of the classifier, as they are independent of each other. The classification task works according to the probability for the occurrence of words in each category. A high posterior probability indicated the high occurrence of the word for the news category. Thus, the news would be categorized into the category which has the highest posterior probability.

Generally, the logistic regression classifiers with different parameters and the SVM classifier with linear kernel, perform well in the text classification tasks, although the performance scores are slightly lower than the Naïve Bayes classifier. Logistic regression can be applied to multiclass classification tasks with the softmax regression approach, which is the generalization of logistic regression [7]. In softmax regression, the input values are normalized into the vector that followed a probability distribution. The softmax function is usually used to derive the losses that can be expected based on the computation of negative log-likelihood during training. This step helps to improve the current training set in order to increase the likelihood of predicting the correct word.

On the other hand, there is a large difference in performance scores between the SVM with linear kernel and polynomial kernel, which is also evident in a study for the comparison of the accuracy between different SVM kernel functions in text classification tasks [8]. As most of the text classification problems are linearly separable, the SVM with linear kernel able to perform better than other kernel functions. On the other hand, the poor performance observed in SVM with the polynomial kernel might be due to its flexible decision boundary. A more flexible decision boundary might result in a high misclassification rate, especially in a linearly separable text classification problem.

In experiment 2, the effect of different text pre-processing steps on the classifier performance is studied. For Naïve Bayes, the F1-score for before stopword removal (Textpre_1), after the stopword removal and stemming (Textpre_3), and after stopword removal and lemmatization (Textpre_4) are similar, yet the F1-score after stopword removal (Textpre_2) is lower than the others. A similar outcomes were also observed in a study by Jayashree, Murthy and Anami that worked on text classification in the Kannada language [9]. The authors concluded that a classifier could make a sentence more understandable with the help of stopwords. Thus, the classifier performance might reduce after the stopwords removal.

The F1-scores for the two logistic regression classifiers and SVM classifier with linear kernel, have been improved for all the text pre-processing steps. However, the F1-score in these classifiers, for stopwords removal and stemming steps are lower as compared to the score in after stopword removal only. In short, the stemming step resulted in slight reduction in performance score after the stopword removal. In contrast, the SVM with polynomial kernel has poor performance for all different pre-processing steps. As mentioned earlier, the possible reason for its performance could be due to the flexibility of its decision boundary. Thus, the pre-processing steps are unable to provide any improvement in the classifier's performance.

In experiment 3, the effect of different n-gram orders on the classifier's performance is studied. The F1-score for all classifiers based on analysis with unigram is the highest, whereas it is the lowest for analysis with trigram. The possible reason for the poor performance in analysis with bigram and trigram is data sparsity, which is indicated by the far lower bigram and trigram occurrence frequency than unigram. In this case, the misclassification rate would be higher in both cases of bigram and trigram, as the occurrence count observed based on the training data is too low. Based on Table XII, the resulting overlapped words are the words we can find in any domain category of BBC news articles, which may cause the news article domains cannot be distinguished based on these words. Thus, it is suggested that a further study can be conducted to study the effect of stopword removal on the classifier's performance after adding the overlapped words into the stopwords list.

IX. Conclusion

For the text classification of BBC news articles in this study, the N-gram analysis is applied on several machine learning algorithms, which are SVM, Naïve Bayes and logistic regression. The BBC news articles dataset has been pre-processed in order to produce an optimal classifier model for text classification. Based on the study, the Naïve Bayes classifier is the best-performing model with the highest F1-score among all classifiers, which is 97.98 %. A further study to determine the effect of different text pre-processing steps on the classifier's performance is conducted. Each classifier is trained and evaluated using data that have gone through different pre-processing steps. The classifier performance before the stopword removal for dataset is compared with the performance after the stopword removal, after the stopword removal and stemming, and after the stopword removal and lemmatization. Similar to the first part of the study, the Naïve Bayes classifier has the best performance among all classifiers. However, it is evident that different text pre-processing steps can result in either improvement or degradation in the classifier's performance. Moreover, the effect of different N-gram orders (unigram, bigram and trigram) on the classifier's performance is also studied. Based on the study, the classifier's performance tends to degrade as the N-gram order goes higher. The possible reason for this occurrence could be the data sparsity, in which the frequency of word occurrence in high N-gram order tends to be lower, that subsequently caused the high risks of misclassification as the classifier.

References

[1] Q.Li, H.Peng, J.Li, C.Xia, R.Yang, L.Sun, P.S.Yu and L.He. "A Survey on Text Classification: From Traditional to Deep Learning," *ACM Trans. Intell. Syst. Technol*, vol. 37, no. 4, pp. 39, 2021.

[2] X.Wang, H.Liu, Z.Yang, J.Chu, L.Yao, Z.Zhao, and B.Zuo, "Research and implementation of a multi-label learning algorithm for Chinese text classification," in *2017 3rd International Conference on Big Data Computing and Communications*., Aug.2017, pp. 68-76.

[3] X.Luo. "Efficient english text classification using selected machine learning techniques," *Alexandria Engineering Journal*, vol. *60, no.* 3, pp. 3401-3409, 2021.

[4] F.Fanny, Y.Muliono and F.Tanzil. "A comparison of text classification methods k-NN, Naïve Bayes, and support vector machine for news classification," *Jurnal Informatika: Jurnal Pengembangan IT*, vol. 3, no. 2, pp. 157-160, 2018.

[5] Patra and D.Singh. "A survey report on text classification with different term weighing methods and comparison between classification algorithms," *International Journal of Computer Applications*, vol. 75, no. 7, 2013.

[6] L.S.Ting, W.H.Ip and A.H.Tsang. "Is Naive Bayes a good classifier for document classification," *International Journal of Software Engineering and Its Applications*, vol. *5*, no.3, pp.37-46, 2011

[7] P.Deng, H.Wang, S.J.Horng, D.Wang, J.Zhang, and H.Zhou, "Softmax regression by using unsupervised ensemble learning," in *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP).*, Dec.2018, pp. 196-201.

[8] N.Kalcheva, M.Karova, and I.Penev, "Comparison of the accuracy of SVM kemel functions in text classification," in *2020 International Conference on Biomedical Innovations and Applications (BIA).* Sep.2020, pp. 141-145.

[9] R.Jayashree, K.S.Murthy, and B.S.Anami. "Effect of stop word removal on the performance of naïve Bayesian methods for text classification in the Kannada language," *International Journal of Artificial Intelligence and Soft Computing*, vol. 4, no. 2-3, pp.264-282, 2014.