Twitter Sentiment Analysis

Introduction

Twitter is one of the popular Social Networking Service (SNS) platforms and people express their opinion with text called tweets. The main purpose of this paper is to analyse tweets' sentiment by using machine learning and deep learning algorithms such as Multinomial Naïve Bayes (NB) and Bidirectional Encoder Representations from Transformers (BERT). For preprocessing of text, removing unnecessary words, balancing the dataset and one-hot encoding is implemented. Especially, the different method is applied for two main algorithms. For Multinomial NB, the Bag of Words (BOW) approach and Term Frequency-Inverse Document Frequency (TF-IDF) is performed. For BERT, tokenization is applied before training. Lastly, all the prediction is evaluated by precision, recall, F1-score and confusion matrix.

Data Exploration

In this paper, the text dataset is from twitter including the review of airlines and its sentiment. To be specific, as Fig. 1. shown, the text dataset given consists of the tweet' id, tweets and its sentiment. Especially, sentiment is a categorical class having 3 classes such as negative, neutral and positive. Also, tweets consist of not only pure text but also tags such as airline's name and links.

	tweet_id	text	airline_sentiment
0	569179849518161920	@united you're good. Thank you!	positive
1	569835751275433984	@AmericanAir way to ruin a vacation, my brothe...	negative
2	568588936852799488	@JetBlue yes thankfully! Catering just got her...	positive
3	569525116725567491	@USAirways The automated message isn't helpful...	negative
4	568807823187976193	@JetBlue I'm #MakingLoveOutofNothingAtAll on m...	positive

Fig. 1. Text Dataset including Tweets and Sentiments

Since the sentiment has 3 different classes, the distribution of all classes is checked. As Fig. 2. shown, the dataset given is imbalanced having the most data in negative class among three classes.

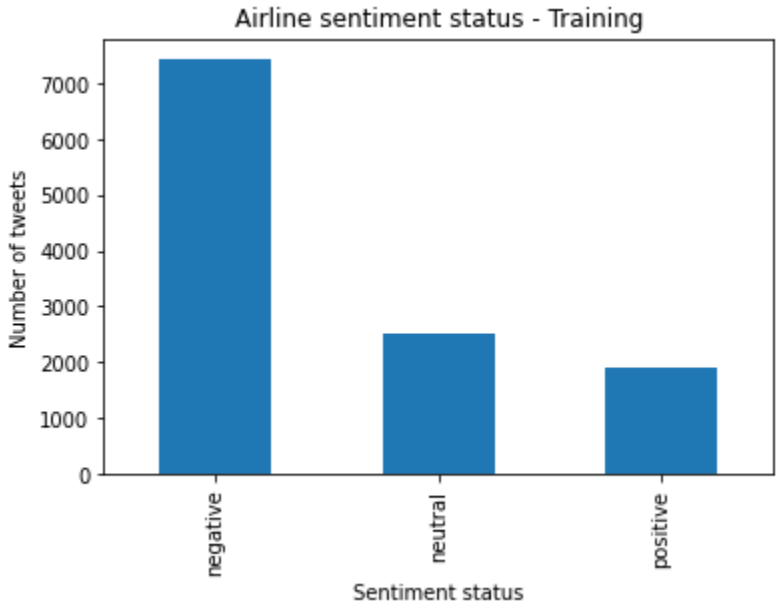


Fig. 2. Distribution of Sentiment in Training set

Methodology

Data Preprocessing

As mentioned earlier, tweets data need to be clear since it contains unnecessary words. For instance, the account tag, link and stopwords can be removed. Especially, since stopwords define no meaning so it has no impact on the sentiment, it can be removed [1]. To remove tagging and links, regular expression is used to detect its form and to remove stopwords, tokenization is applied to the tweet to compare with defined stopwords from NLTK library [2].

Additionally, as the data exploration shows that the dataset is imbalanced, it needs preprocessing to avoid overfitting [3]. Especially, the amount gap between negative class and others can reduce the model's performance. Thus, resampling is required to balance the dataset such as upsampling and downsampling. However, downsampling makes the dataset smaller and the small dataset can cause poor performance. Therefore, upsampling is implemented using the *RandomOverSampler()* method [4]. This method will choose existing data randomly so that the randomness can consist in oversampled dataset. However, upsampling has been conducted only on training set to avoid the risk of overfitting. Since upsampling is iterating the data, oversampled validation and test set can result in overfitting.

Moreover, to improve the performance, the categorical class – sentiments – has been converted into numerical class such as 0, 1 and 2. Not only that, one-hot encoding has been adopted to these numerical classes.

Architecture

Naïve Bayes algorithm is one of the state-of-art models for Natural Language Processing (NLP). In particular, Multinomial Naïve Bayes (NB) algorithm is chosen since it has proven its performance with multiple text classes by Xu *et al.* [5]. Before conducting the Naïve Bayes method, the Bag of Words (BOW) approach and Term Frequency-Inverse Document Frequency (TF-IDF) method is conducted. BOW analyses

the frequency of each word in text and then TF-IDF treats the low-frequency word as an important word in the corpus [6]. Additionally, hyper-parameter tuning is not applied for Multinomial Naïve Bayes since it only has one parameter to change.

Furthermore, Bidirectional Encoder Representations from Transformers (BERT) is also implemented since it is one of the most popular NLP deep learning models. BERT processes all tokens before and after for each token of input text by using the transformer encoder [7]. In other words, it helps machine to understand the natural language using surrounded text as a context. Thus, BERT outperforms with text classification in many studies [8], [9]. Before applying BERT, tokenization is implemented with BERT's own tokenization method, *BertTokenizerFast()*. Since the BERT method achieves 0.80 accuracy at first epochs, hyper-parameter tuning was not necessary.

For evaluation, precision, recall and F1-score is generated for both Multinomial NB and BERT by using the *classification_report()* method [10]. These methods are based on the prediction result such as True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). Precision, recall and F1-score are calculated by

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2}$$

Particularly, F1-score is a proper measure for imbalanced dataset rather than precision and recall. Since the test dataset given is distributed unevenly throughout the classes, F1-score would be the appropriate evaluation.

Results

For the evaluation, precision, recall and F1-score have been generated for both Multinomial NB and BERT. According to Table 4, Multinomial NB shows great performance of 0.87 for class Negative, on the other hand, it shows lowest accuracy for Neutral class of 0.60 F1-score. It is suspected that Neutral sentiment is ambiguous so that it has a limitation to predict accurately with machine learning method.

Table I: Precision, Recall and F1-score of Multinomial Naïve Bayes

	Precision	Recall	F1-score	Support
Negative	0.87	0.86	0.87	918
Neutral	0.61	0.59	0.60	310
Positive	0.65	0.73	0.69	236
Accuracy	-	-	0.78	1464
Macro Average	0.71	0.73	0.72	1464
Weighted Average	0.78	0.78	0.78	1464

However, the BERT model results in higher performance than Multinomial NB as expected earlier. As Table 5 displayed, F1-score for each class is 0.88, 0.64 and 0.73 which are all higher than the result of Multinomial NB. However, the Neutral class has the lowest F1-score of 0.64 so it is suspected that Neutral sentiment is difficult to classify even with context.

Table II: Precision, Recall and F1-score of BERT

	Precision	Recall	F1-score	Support
Negative	0.86	0.90	0.88	918
Neutral	0.70	0.59	0.64	310
Positive	0.71	0.75	0.73	236
Micro Average	0.81	0.81	0.81	1464
Macro Average	0.76	0.75	0.75	1464
Weighted Average	0.80	0.81	0.80	1464
Samples Average	0.81	0.81	0.81	1464

Nevertheless, Fig. 3. presented that both Multinomial NB and BERT predicts the class quite well. As the test dataset has the most data in Negative class, it seems like the Negative class has outstanding performance. Still, Neutral and Positive class also got its true value the most compared to their false value.

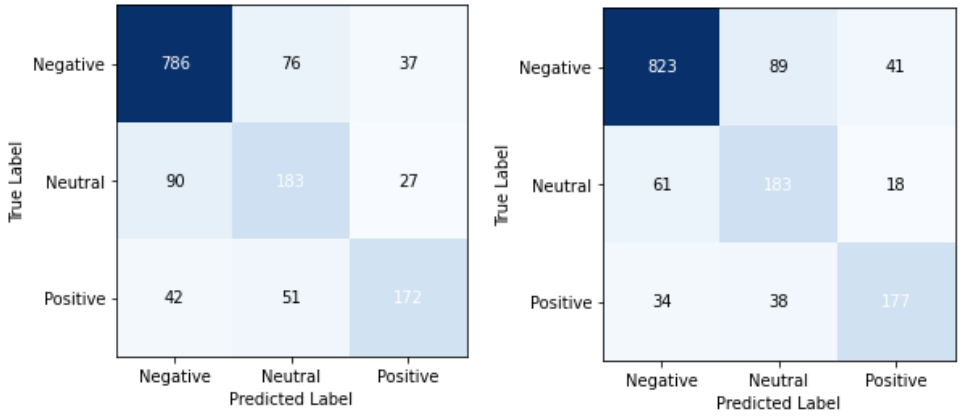


Fig. 3. Confusion Matrix of Multinomial NB (Left) and BERT (Right)

Discussion

There have been essential factors to improve the accuracy of models during experiments. Most importantly, preprocessing take the most responsibility to increase all the model's performance. Before fixing the imbalance of training dataset, the result with original data tended to overfitting and predict only one class. To be specific, it always predicts every class as Negative sentiment. Since the Negative class has the most data among all, it causes lazy overfitting. Nonetheless, this problem is solved by balancing the dataset by oversampling. However, since upsampling is iterative, it could cause overfitting with other cases. Thus, it could have done better with using different weighting for each class.

Additionally, cleaning the corpus tweet also helped to improve accuracy and the speed of training. By removing unnecessary words such as tags, links and stopwords, it increases its performance. Still, it is suggested to compare the results with removing more elements such as hashtag. Since hashtag also includes factor of sentiment sometimes, it would be interesting to analyse the result without it.

Moreover, due to the limitation of GPU power, only two algorithms have been implemented. It could have done better with comparing results with more machine learning and deep learning method to get higher performance.

References

[1] S. Teja, "Stop Words in NLP," Medium, 10 Jun 2020. [Online]. Available: https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47. [Accessed 10 Dec 2022].

[2] NLTK, "Stopwords," 2023. [Online]. Available: https://www.nltk.org/search.html?q=stopwords&check_keywords=yes&area=default. [Accessed 10 Dec 2022].

[3] Z. Li, K. Kamnitsas and B. Glocker, "Analyzing Overfitting Under Class Imbalance in Neural Networks for Image Segmentation," *IEEE Transactions on Medical Imaging,* vol. 40, no. 3, pp. 1065-1077, 2021.

[4] G. Lematre, F. Nogueira and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," *Journal of Machine Learning Research,* vol. 18, no. 17, pp. 1-5, 2017.

[5] S. Xu, . Y. Li and Z. Wang, "Bayesian Multinomial Naïve Bayes Classifier to Text Classification," *Advanced Multimedia and Ubiquitous Engineering,* vol. 448, pp. 347-352, 2017.

[6] Y. HaCohen-Kerner, D. Miller and Y. Yigal, "The influence of preprocessing on text classification using a bag-of-words representation," *PLoS ONE,* vol. 15(5), 2020.

[7] M. Abadi, A. Agarwal, P. Barham , E. Brevdo, Z. Chen, C. Citro and et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015. [Online]. Available: https://www.tensorflow.org/. [Accessed 2 Jan 2023].

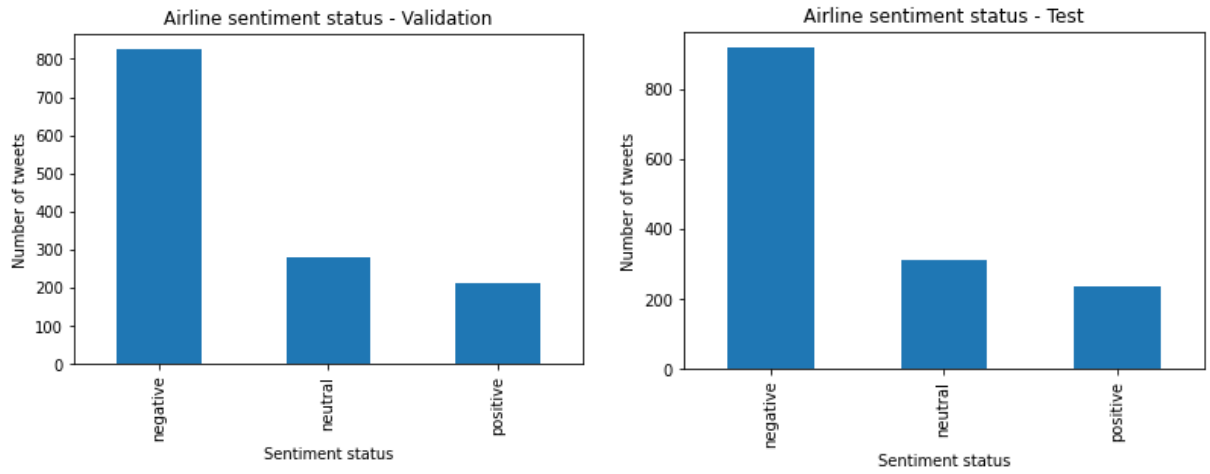
[8] S. González-Carvajal and E. C. Garrido-Merchán, "Comparing BERT against traditional machine learning text classification," vol. 2, 2021.

[9] S. Zheng and M. Yang, "A New Method of Improving BERT for Text Classification," *Intelligence Science and Big Data Engineering. Big Data and Machine Learning,* vol. 11936, p. 442–452, 2019.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel and et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research,* vol. 12, pp. 2825-2830, 2011.

Appendices

Appendix A - Distribution of Sentiment in Validation (Left) and Test (Right) set



Appendix B – Final Model Architecture of BERT

Layer (type)	Output Shape	Param #	Connected to
input_3 (InputLayer)	[(None, 64)]	0	[]
input_4 (InputLayer)	[(None, 64)]	0	[]
tf_bert_model_1 (TFBertModel)	TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, 64, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)	109482240	['input_3[0][0]', 'input_4[0][0]']
dense_1 (Dense)	(None, 3)	2307	['tf_bert_model_1[0][1]']

Total params: 109,484,547
Trainable params: 109,484,547
Non-trainable params: 0