News Headlines Classification

Yilin Yang Huiting Song Shiyu Wang Tianyi Xu

ANLY580

Contents

1 Introduction

News is happening every minute of every day, and people can use it to find out the current happenings, the latest policies issued or the most exciting results of the World Cup competition. The source of those news can be from TV, from people talking to each other, or online. With nearly half a billion visits to Google News each month, it's clear that getting news and information on the web is now a common occurrence. People can search for news keywords to view relevant news. Starting from 2020, coronavirus has become the most common concern due to the rampant of new coronavirus. This is not difficult to observe from the data. Coronavirus tops the list of searches on online platforms and is associated with 70 out of every 1,000 inputs. Today, users typically only need to enter one word or one tag and the algorithm achieves a highly correlative relevant pushing. For news, pinpoint pushing relies on news classification. For example, different news headlines are represented by different labels so that relevant news can be pushed when people search for the label terms. So how to effectively implement accurate classification of news labels and accurate relevance push is the direction we need to study in this project.

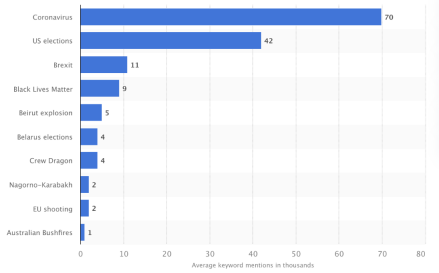


Figure 1: Hot topic during 2020

People always get new information from different sources and news platform is one of the most important sources. We want to know what happened from different sources and it is very important to classify these information and get something relative to extend reading.

Figure 2: Visit click of google news

2 Project Purpose

The purpose of this project is to classfy the news headlines by finding the model with best accuracy. The output of news headlines classification is the return of news labels and relative news. Through modelling with Naive Bayes, SVM, and BERT, we expect to have a comparable model result and best model selection. The model with high accuracy can help us to do the future prediction of news headlines. In the end, Two demos will be created to display the output labels of a random news headline input, and the relevent news according to the classification result.

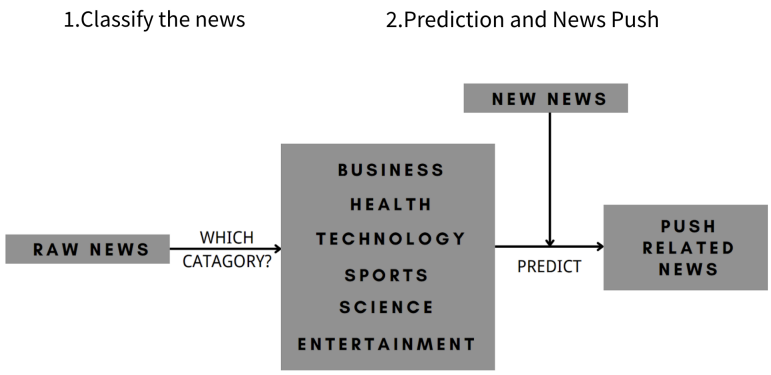


Figure 3: Project Purpose

3 Data Collection

We use python package selenium to crawl the data. Each row has title, topic, subtopic and its topic label.

news title	topic	sub topic	topic label
Holiday shopping returned to a lower-key normal this Black Friday	Business	Latest	0
Elon Musk says Twitter will relaunch its verification program next week	Business	Latest	0
Musk says Twitter will launch blue check subscription next week	Business	Latest	0
Twitter relaunching Verified, with manual authentication checks	Business	Latest	0
Twitter Will 'Tentatively' Relaunch Paid Verification System Next Friday: Musk	Business	Latest	0
...	...	...	...

You can see the distribution of news headlines. We want to make sure that our data are not imbalanced and actually the not-equally distributed data will help us identify the accuracy of different models.

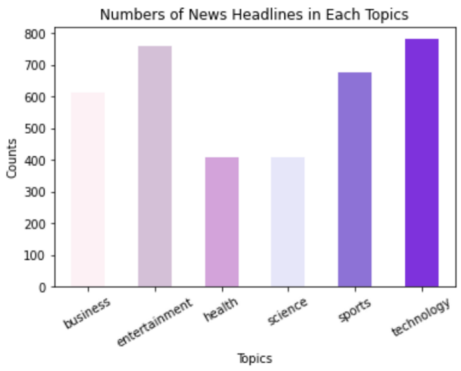


Figure 4: Data Distribution

4 Data Processing: Tokenization

Tokenization is the process of demarcating and possibly classifying sections of a string of input characters. The resulting tokens are then passed on to some other form of processing. Machine Learning algorithms could only accept tensor-shaped data so we need to split the text into words. First, we split the text into sentences and each row points to a sentence. We also collect all words without repetition to create a dictionary and assign each word an index. Therefore we could use a vector of index to refer to a sentence so the computer could deal with it.

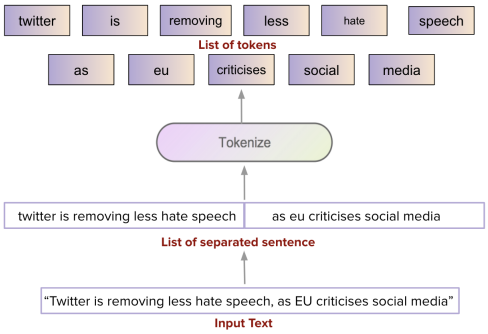


Figure 5: Tokenization

5 Model Building

5.1 Naive Bayes

5.1.1 Illustration

In our project, Naive Bayes classifier calculates the probability of an event in the following steps:

1. Calculate the prior probability for given class labels
2. Find Likelihood probability with each attribute for each class
3. Put these value in Bayes Formula and calculate posterior probability.
4. See which class has a higher probability, given the input belongs to the higher probability class.

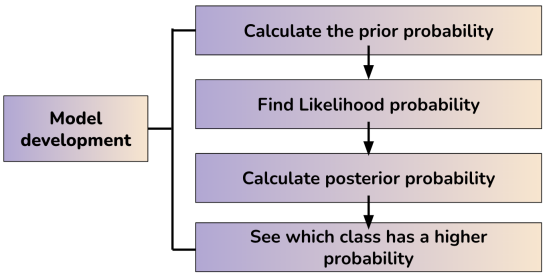


Figure 6: Tokenization

5.1.2 Optimization-Laplace Smoothing

In order to increase the accuracy, we need to do optimization on our model:
The Laplace smoothing is useful for solving the zero probability problem in Naive Bayes algorithm. If the word is absent in the training dataset, then we don't have its likelihood. But we cannot conclude that the probability of the presence of this word is zero. At this time Laplacian smoothing will let us to set a certain alpha number in order to prevent this situation. Laplace smoothing is a smoothing technique that handles the problem of zero probability in Naïve Bayes. Using Laplace smoothing, we can represent $P(notintrainword|label)$ as

$$P(not\,in\,train\,word|label) = \frac{number\,of\,reviews\,with\,not\,in\,train\,word\,and\,label = label + \alpha}{M + \alpha * K}$$

Here,
α represents the smoothing parameter,
K represents the number of dimensions (features) in the data, and
N represents the number of reviews with y = label that we want.
In our project, through several attempts, we set alpha = 0.6 because of the higher accuracy of the final result. There will be overfit if we have so much zero-counted words; Since our dataset is sufficient, we don't need unlabeled data to improve accuracy. At this time, low frequencies words can only be served as noise and decrease the accuracy.

5.1.3 Final results

Most of accuracy of different labels is more than 75%. Precision and Recall are also robust, which do not have abnormal situation. The whole accuracy is 79% and we believe it is a good model to classify news.

```
print(classification_report(gold_label, pre_label))
✓ 0.3s
```

	precision	recall	f1-score	support
0	0.76	0.77	0.77	136
1	0.74	0.83	0.78	149
2	0.77	0.86	0.81	140
3	0.92	0.86	0.89	148
4	0.75	0.67	0.71	72
5	0.77	0.59	0.67	86
accuracy			0.79	731
macro avg	0.79	0.76	0.77	731
weighted avg	0.79	0.79	0.79	731

Figure 7: Final results

By checking the confusion matrix of naive bayes, most of observations are located on the diagonal of the matrix. The number on the heatmap is not actual number of observations but the relatively importance of each cell.

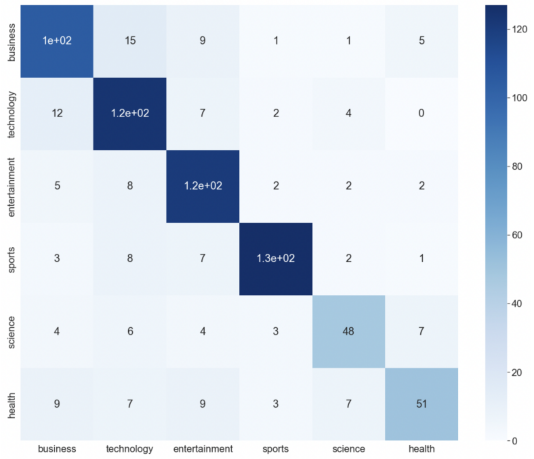


Figure 8: Confusion Matrix

5.2 Support Vector Machine

5.2.1 Illustration

In this part, we tried Support Vector Machine(SVM) to make classification on news headlines. First, we extract features from original data set after tokenization. Here we used a pre-trained BERT model to convert original tokens to new features. Second, we build a svm model and use classification report, accuracy

score and confusion matrix to get model evaluations. Third, according to preliminary results, we did hyper-parameter tuning to improve model performance. Important parameters in SVM model of the scikit-learn package are C, kernel, degree, gamma, etc.

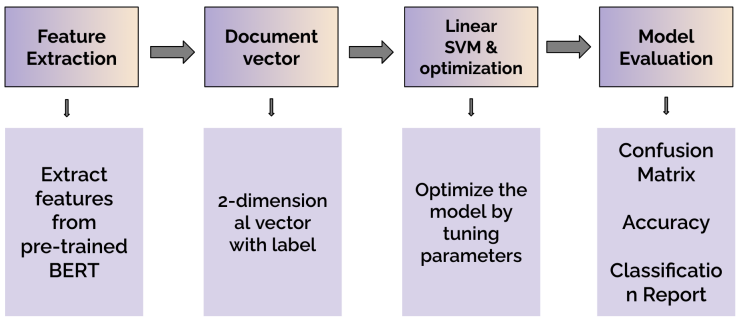
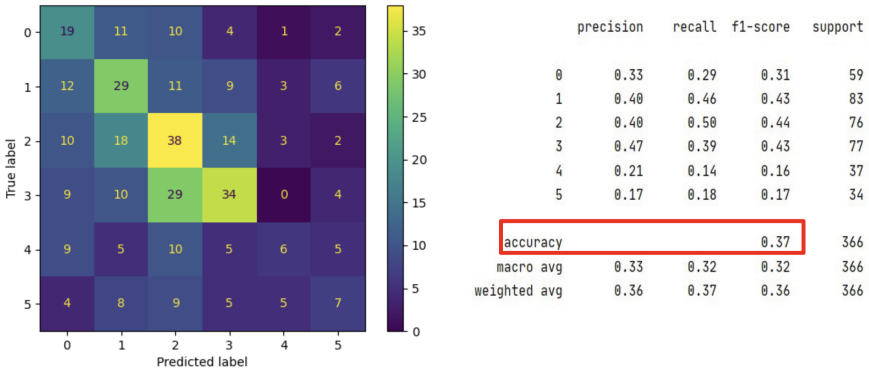


Figure 9: Illustration

5.2.2 Result

Overall, the model performance of SVM was not satisfying. According to the confusion matrix in figure 9, the x-axis showed classified results of predicted labels and y-axis showed that of true labels. Also in classification report in figure 9, we saw the classification accuracy was only 0.37 which means in test set, there were only 37% data were correctly classified. However, this was the best model performance after finely tuning the parameters.



	precision	recall	f1-score	support
0	0.33	0.29	0.31	59
1	0.40	0.46	0.43	83
2	0.40	0.50	0.44	76
3	0.47	0.39	0.43	77
4	0.21	0.14	0.16	37
5	0.17	0.18	0.17	34
accuracy			0.37	366
macro avg	0.33	0.32	0.32	366
weighted avg	0.36	0.37	0.36	366

Figure 10: Result

5.2.3 Conclusion

Generally speaking, the overall accuracy of SVM model is very low. We concluded several reasons of the bad performance after research and discussion.

1. When extracting features, the BERT model needs to be fine-tuned but we didn't.

2. When dealing with high dimension features, SVM could not handle it accurately. Moreover, here we summarized pros and cons of the SVM model on our dataset. The advantage was that we used BERT model to extract features. This was an inovation point of our project although it was not satifying. The disadvantage was that it has very low accuracy, hence there is no value for analyzing the result furthermore.

5.3 BERT

5.3.1 Encoding

Now, we use pre-trained BERT model. BERT is an open source machine learning framework for natural language processing (NLP). It is designed to help computers understand the meaning of ambiguous language in text by using surrounding text to establish context. The BERT framework was pre-trained using text from Wikipedia and can be fine-tuned with question and answer datasets. BERT can be used in classification task and next prediction task. Here, the main aim of using the model is to categorize a text into one of the predefined labels.
BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection. (In NLP, this process is called attention.)
First, we will need to do the tokenize encoding. Since we have tokenize all the sentences in the data processing, so in this process, we group the tokens in list of tokens to make a process sentences. to preprocess the input text data, the first thing we will have to do is to add the [CLS] token at the beginning, and the [SEP] token at the end of each input text. The BERT model receives a fixed length of sentence as input. The max length is 64. For sentences that are shorter than this maximum length, we will have to add paddings (empty tokens) to the sentences to make up the length. Then, we can encode the tokens to get the inputs id and attention mask.
The "attention mask" tells the model which tokens should be attended to and which (the [PAD] tokens) should not

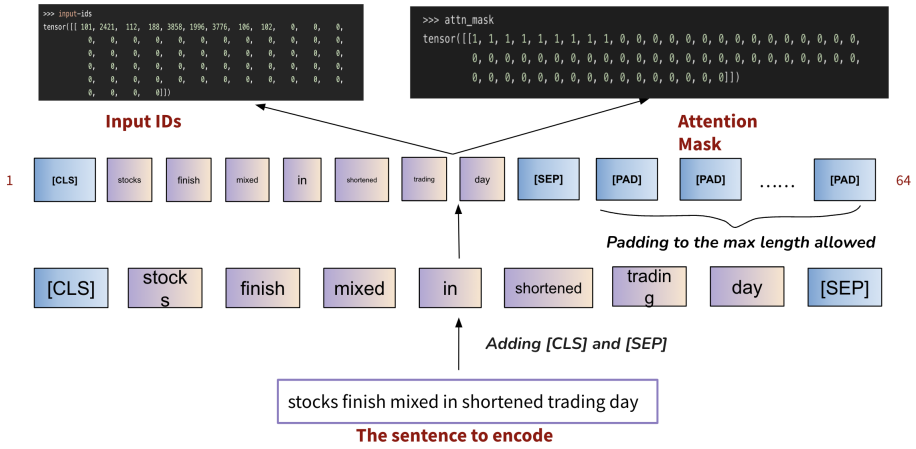


Figure 11: Encoding

5.3.2 Modeling

In the modelling process, we use we use a pre-trained Bert for sentence classification from bert-base-uncased model.If we input a sentence, model will return us the logits which is the probability of belonging to a label. Optimize our model. We train the model for 70 epochs and we use Adam as the optimizer.
(Adam algorithm : Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data. This algorithm is used to accelerate the gradient descent algorithm by taking into consideration the 'exponentially weighted average' of the gradients. It is very straightforward to implement and computationally efficient)
Using averages makes the algorithm converge towards the minima in a faster pace. We also need to use categorical cross entropy as our loss function since we're dealing with multi-class classification. Since the bert model return logits, within each iteration we get the loss as return. After that, we can use the test data to evaluate the model's performance on unseen data. And we set the process return us the accuracy and the confusion matrix.

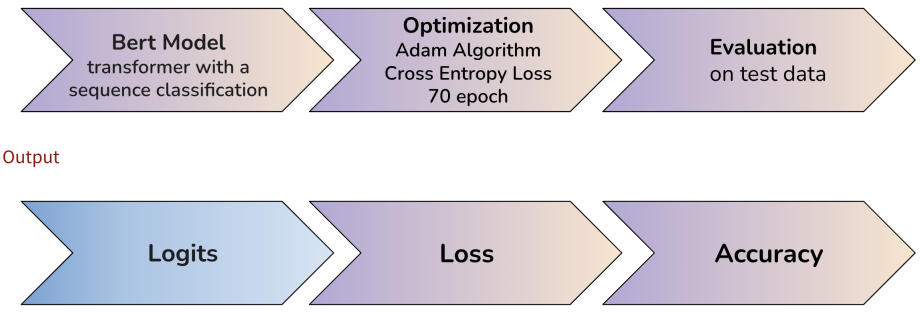


Figure 12: Modeling

5.3.3 Result

After training the model, The train loss and accuracy of test date was recorded. Within 70 epochs' return, the number 54 epoch output the highest accuracy. The accuracy reaches 86%. This indicates that BERT model can provide highest 86% of accuracy on our dataset and has the highest 86% possibility to provide us the accurate labels of news headlines. By grabbing the result from confusion matrix, most of the news headlines labels BERT model predicted can match with the original labels.

```
[epoch 54] train_loss: 0.026  dev_accuracy: 0.860
              precision    recall  f1-score   support

           0       0.81      0.83      0.82       123
           1       0.82      0.87      0.85       162
           2       0.91      0.86      0.89       148
           3       0.98      0.94      0.96       127
           4       0.86      0.86      0.86        88
           5       0.80      0.80      0.80        83

    accuracy                           0.87       731
   macro avg       0.86      0.86      0.86       731
weighted avg       0.87      0.87      0.87       731
```

Figure 13: Result

```
[[102  11   4   0   2   4]
 [ 12 141   4   0   3   2]
 [  6   9 128   3   1   1]
 [  0   1   3 120   0   3]
 [  1   4   0   0  76   7]
 [  5   5   1   0   6  66]]
```

	Correctly Predicted Counts	Error Counts
Business	102	21
Technology	141	21
Entertainment	128	20
Sports	120	7
Science	76	12
Health	66	17

Figure 14: Confusion matrix

6 Comparison

Through comparing the accuracy of Naive Bayes, SVM, and BERT model, BERT model returns the highest accuracy as 86% and has relatively robust recall and precision for all topics. SVM model has the lowest accuracy due to the dimension limitation. The BERT transformer encoder reads the entire sentence input at once as it is bidirectional. Its bidirectional characteristic allows the model to learn all right and left of the word to have a deeper understanding of language context and flow. Therefore, the best model can be used to classify the news headlines is BERT model based on our dataset.

7 Prediction

The final prediction can be concluded into the following steps. The Raw text is imputed for the preprocess, then we use tokenize to split the sentence word by word.We create dictionary and assign index to words. Then we evaluate the model and save the best model. As we can see the best model in our project is Bert. Then we use Bert to calculate the class probabilities and choose the class with highest probability for classification result. If we input "Twitter is removing less hate speech, as EU criticises social media", the output will be business for classification.

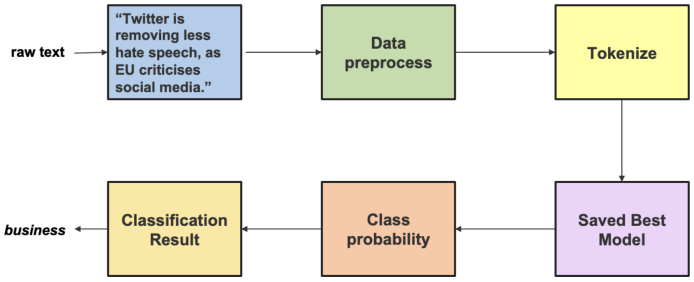


Figure 15: Prediction

8 Demo

8.1 Demo-1

We made a interactive demo for the BERT model. We input any news headlines in the text bot, it will predict a label for us. We randomly copy news headlines on the Internet and it turns out that nearly all the text are correctly classified.

Figure 16: Demo-1

8.2 Demo-2

Our second demo is that after the label prediction, we could generate related news links recommendation.

Figure 17: Demo-2

9 Conclusion

Now we have a whole working process. We use bert model to classify the news headline with 86% acc. When we have new news, we could also predict its classification and push related news for reading. You could get the recent news title and relative links.

Although we have 3 types of model and have 86% of accuracy with BERT, in real practice we need a more precise model to get higher accuracy. Some initial problems will also effect our classification. Sometimes a piece of news could be classified to two topics because it conclude different contents. In further learning and research, we need to keep learning and update our model to make our accuracy higher.