

DM2023 Lab2 Report

Introduction

There is a dataset which was crawled from Twitter, and has already labeled the emotion for these tweets by some specific hashtags in the original text. There are 8 emotions in the dataset: anger, anticipation, disgust, fear, sadness, surprise, trust, and joy. The final goal is to learn a model that is able to predict the emotion behind each tweet.

Methodology

1. Data setting:
 - First, read the dataset.
 - Second, handle data extraction and manipulation from a JSON object.
 - Third, separate the training data and testing data.
2. Data preprocessing(Clean text):
 - **Remove punctuation and special characters:** Uses regular expressions to replace or remove punctuation and special characters from the text.
 - **Convert to lowercase:** Transforms all text into lowercase, ensuring uniformity in text formatting.
 - **Remove numbers:** Uses regular expressions to eliminate numbers from the text.
 - **Remove stop words:** Utilizing the set of stop words from NLTK, it tokenizes the text and removes stop words
3. Model training: Utilizes the Hugging Face Transformers library to perform sequence classification tasks using the BERT model.
 - **Initialize BERT tokenizer and model:** Uses the `bert-base-uncased` model, which is a pre-trained variant of the BERT model. The tokenizer converts text into a format that the model understands, and `TFBertForSequenceClassification` is the BERT model used for sequence classification.
 - **Encode text data:** If the text surpasses the specified ``max_length``, it gets truncated to match that length. Additionally, padding is applied when the sequence falls short of the ``max_length``. This ensures that all sequences have the same length, as models often expect fixed-length input sequences.
 - **Compile the model:** Compiles the model using the Adam optimizer. It utilizes Sparse Categorical Cross entropy as the loss function for multi-class classification. Accuracy is used as the evaluation metric.
 - **Training the model:** The model is trained using the training data for two epochs, with a batch size of 32.

Experiments and Results

I have tried some other models, so I will compare each method.

Feature Engineering:

1. Time: Separate the time into year, month, day, hour, and transform the hour to cosine.
2. Hashtags: Transform it by label encoding.
3. Text:
 - Clean text: Same progress as the data preprocessing above.
 - Extract keywords: Utilize `word_tokenize` function.

- Word2Vec: Utilize a Word2Vec model, setting it to represent words in a 100-dimensional vector space, training within a context window of size 5, considering all words that appear at least once, and using 4 CPU cores for the training process.
- TF-IDF: Convert the cleaned text into a matrix represented by TF-IDF, and use SVD to perform dimensionality reduction operations.

Model training:

1. XGBoost with n_estimators: 100
2. Random Forest with n_estimators: 100
3. LGB with metric: 'multi_logloss', num_leaves: 31, learning_rate: 0.05
4. Ensemble model: Random Forest with n_estimators: 200, AdaBoostClassifier with n_estimators: 200 and XGBoost with n_estimators: 100, utilizing soft voting to ensemble
5. RNN: An embedding layer that maps a vocabulary of size 2000 to 16-dimensional vectors. Following there is an LSTM layer with 64 LSTM units, succeeded by two fully connected layers with 128 and 64 neurons, using ReLU activation. The final layer is a softmax dense layer with 8 neurons for multi-class classification. The model utilizes categorical cross-entropy loss, Adam optimizer, and employs a custom callback to halt training when validation accuracy surpasses 90%. Training is performed over 15 epochs with a batch size of 32, validated against test data.

Score	TF-IDF + LGB	TF-IDF + Random forest	TF-IDF + XGBoost	W2V + Ensemble	RNN	Final Model
Public	0.35123	0.35385	0.40241	0.43675	0.24855	0.50682
Private	0.34661	0.35061	0.39007	0.42442	0.24774	0.4955

Table 1. compare the score of all the models I've utilized.

Some thoughts:




For all the models I've utilized, I don't really try a lot of hyper parameters against each model, so maybe this is a part that I can improve more. For the RNN model, I think it came across overfitting, since the accuracy calculated by myself is basic around 60%, but kind of running out of time, so I don't have the time to deal with the problem.

As for the final model I chose, utilizing a pre-trained BERT model, it's been running for extended periods, making it challenging to fine-tune parameters effectively. Anyway, I'm still having a really interesting time with this lab2.

Ps 因為電腦在跑 tensorflow 的時候會出問題，所以剛開始寫作業的時候是使用 google colab 來完成 First part，但是後來因為 google colab 跑的時候有點慢加上在跑的時候一直需要網路，有點不方便，所以後來使用 Kaggle notebook 進行 Second part，我有把這個跑出來的結果下載下來，在 [DM2023-Lab2-Final model.ipynb](#)。同時我也有把最後 model 的 code (Second) 放上 [DM2023_Lab2_Homework_colab.ipynb](#)。

總的來說，[DM2023_Lab2_Homework_colab.ipynb](#) 上有完整的程式碼（需要先把 data 放上雲端），但是如果想看 Second part 的每個 cells 的 output 就需要看 [DM2023-Lab2-Final model.ipynb](#)。真的非常抱歉這麼麻煩，感謝。

另外，我怕截圖跑不出來，在這邊也貼上，感謝。

Submission and Description		Private Score ⓘ	Public Score ⓘ	Selected
	submission.csv Complete · 1d ago	0.4955	0.50682	<input type="checkbox"/>
	DM2023 ISA5810 Lab2 Homework In this competition, we provide a dataset which was crawled from Twitter, and we have already labeled the emotion for these tweets. Community · 99 Teams · Private · 8 minutes ago			26/99  ...