Bert-based Model for Classifying Suitable English Sentences for Bilingual Children

Ma, Sum Yi (symaab@connect.ust.hk)
Sam, Fetullakh (fsam@connect.ust.hk)

Introduction

Recently, it has become common to see parents complaining about the inappropriate content of books for children. People are concerned and blame these books for the harmful moral values promoted in them. They tried to throw these books away to keep their children safe, but the problem will never end if only these actions are taken (張雅婷, & 杜潔心, 2016) (白毅鵬,2020). Another issue is that some books are too advanced for Hong Kong children due to the absence of a Hong Kong official age-rating system on books. It could especially confuse bilingual children who are not a native English speaker and eventually make them uninterested in reading books. The root cause, the lack of author reviewing, must be addressed.

Thus, using machine learning to classify whether a sentence in the story is suitable for bilingual children to read could facilitate authors to review their work effectively. Then, the author could focus on their writing task instead of reviewing their text tediously. For example, an author could input a sentence into the AI model we proposed, and the model would identify whether the sentence contains inappropriate moral values, swear words, or advanced words that children most probably can't understand. If yes, the model would tell the author whether it should be changed to be suitable for child stories. In such a case, the model would act as an assistant that suggests to the author whether the author should edit the sentences written.

Dataset

Basic information

The fundamental dataset used is [the hate speech dataset] available by Kaggle. The dataset contains 24783 entries containing features related to hate speech recognition, such as count of hate speech, count of offensive language, etc. To slightly reduce the data preprocessing time, we select the first 20000 entries of it.

The details of the dataset are in the following:

Column Name	Data type	Description
count	Integer	The total number of annotations for each tweet.
hate_speech_count	Integer	The number of annotations classifying a tweet as hate speech.
offensive_language_count	Integer	The number of annotations classifying a tweet as offensive language.
neither_count	Integer	The number of annotations classifying a tweet as neither hate speech nor offensive language.
class	Integer, (0,1,2)	Class 0 : neither hate speech nor offensive language Class 1 : offensive language Class 2 : hate speech
tweet	string	a sequence of words (string) that is short in length. If we do not consider special content

		such as username. The maximum length of the tweet content is 280 characters

Preprocessing

However, this dataset is not sufficient for us to complete the task of classifying suitable sentences in English for bilingual children. We need to apply several preprocessing techniques to it.

Merging with another hate speech dataset

The dataset above is imbalanced. It has 19190 data entries in class 1, and 4163 data entries in class 2, but only 1430 data entries in class 0. The major class 1 occupies around 78% of the entire database while class 0 only occupies around 6%. To improve the data balance while not making up the data by data augmentation, we merged it with another hate speech recognition dataset with 87% non Hate speech data entries.

Here are the details of that dataset:

Column Name	Data type	Description
file_id	string	The id of the corresponding text file.
user_id	string	The id of the user who posts the sentences in the text file.
subforum_id	Integer	The id of the subforum that the sentences are posted on.
#num_context	Integer	Number of context
label	string,(noHate,hate,relation,idk/skip)	Whether the file contains hate speech.

Since the text in Class 1 and Class 2 in the first hate speech dataset are both not suitable for children to read, we could view it as a binary label: Class 1 and Class 2 represent speech that contains immoral values, while Class 0 is normal text. Then, we do not need to care about the label inconsistency between the first and second datasets. We only need to convert 'noHate' and 'hate' into Class 0 and Class 1 respectively, and drop the data entries in the "relation" and "idk/skip" classes. We also drop the columns irrelevant to our task and replace the column 'file_id' with the content inside the file as well.

Therefore, we gained a new dataset with the details below:

Column Name	Data type	Description
immoral text	Integer,(0,1,2)	Class 0: text that does not contain immoral values Class 1: hate speech Class 2: offensive language Noticing that both Class 1 and 2 would harm the children.
tweet	string	a sequence of words (string)

After the merging, we have 35% data entries in class 0 and 80% data entries in class 1 or class 2. Compared with that of the original dataset, the data imbalance is improved.

Feature engineering

When we are building the classifier, we also need to consider the readability of the sentences, but not only consider whether the sentences contain offensive language or hate speech which might affect the mental development of children. Hence, we decided to add a new feature to the dataset: readability score, which indicates the percentage of known English words for bilingual children in a particular sentence.

Hence, we found a bilingual children's speech corpus and tokenized it by words to form a set of English words that bilingual children could understand. Then, we tokenize the 'tweet' column similarly. Next, we count the number of tokens that exist in both column 'tweet' and the bilingual children's speech corpus, α, for every data entry.

Let the total number of tokens in column "tweet' for a data entry is χ:

$$Readability\ Score\ = \frac{\alpha}{\chi}$$

Then, the edited dataset would have the following details:

Column Name	Data type	Description
immoral text	Integer,(0,1,2)	Class 0: text that does not contain immoral values Class 1: hate speech Class 2: offensive language Noticing that both Class 1 and 2 would harm the children.
tweet	string	a sequence of words (string). It may contain special characters.
readability score	float	the percentage of known English words for bilingual children in a tweet.

Relabelling

Because of the newly added feature, we have to relabel the dataset considering both the readability score and whether the text is immoral.

A research (Schmitt, Jiang, and Grabe, 2011) indicates that 90% of vocabulary coverage could reach a 50% comprehension of text, a 95% of vocabulary coverage could reach 60% comprehension. Considering that the 3000-vocabulary bilingual children corpus does not cover every domain of words that children may know, we decided to take the threshold of 90% on the readability score. Now, if a text is suitable for children to read, it should have a readability score equal to or more than 0.9 and have a '0' in the column 'immoral text'.

Now we have a final column structure of the dataset:

Column Name	Data type	Description
immoral text	Integer,(0,1,2)	Class 0: text that does not contain immoral values Class 1: hate speech Class 2: offensive language Noticing that both Class 1 and 2 would harm the children.
tweet	string	a sequence of words (string). It may contain special characters.
readability score	Float, [0,1]	the percentage of known English words for bilingual children in a tweet.

Class	Integer, either 0 or 1	Class 0:not suitable for children to read Class 1:suitable for children to read

Undersampling

After the relabelling, we have to check the imbalance situation again. The latest dataset above has 717 data entries in class 1, while all the remaining samples are in class 0. There is an extreme imbalance. Therefore, undersampling is needed to handle this situation.

We pick 717 samples from the samples in Class 0 with all the samples in Class 1 to create the final dataset. Now, our dataset has 50% data in Class 0 and Class 1, respectively.

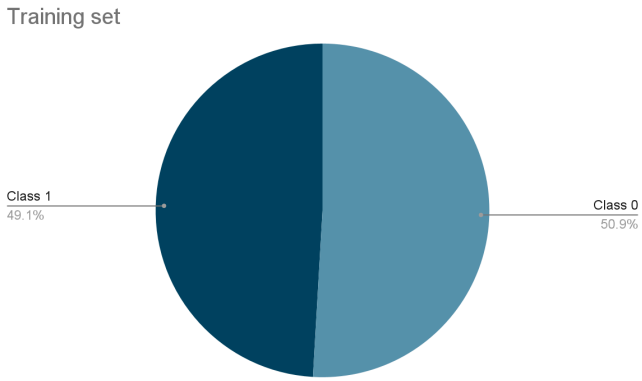
Data Cleaning

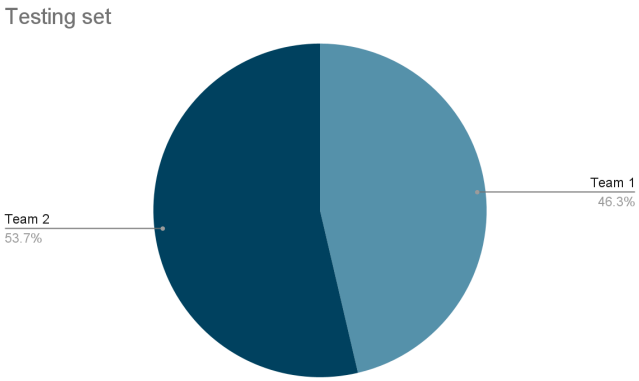
The text in column 'tweet' may contain special characters such as '@','&','#', and special content such as URL, but sentences in children's books would not contain such elements. Hence, we have to clean up the text by removing those special characters and special content.

Bert Tokenization

We tokenize the text in column 'tweet' using the bert-base-uncased tokenizer from the Hugging Face library before training. According to the Hugging Face documentation(2001), The texts are lowercase and tokenized using WordPiece and a vocabulary size of 30,000.

Train-test set splitting





After shuffling the final dataset, we split the dataset into a train set and a test set. The shuffling ensures the randomness of the order or the data entries, while also keeping an acceptable balance of the data.

Considering the small size of our dataset, we make the train set to have 80% of the data entries and the test set to have 20% of the data set, to ensure sufficient data for training.

Machine Learning Task

Two main ML tasks performed on the dataset: tokenizer for turning input text into tokens suitable for classifying and binary classification itself that identifies whether a sentence of tokens is appropriate for bilingual children.

BERT Tokenizer

Input: Raw text (sentences or paragraphs).

Output: Tokens (a sequence of subwords or characters representing the input text).

Description: BERT Tokenizer breaks down words in raw text into sub-words or characters to allow the model to understand the context of the text more effectively.

Binary classifier

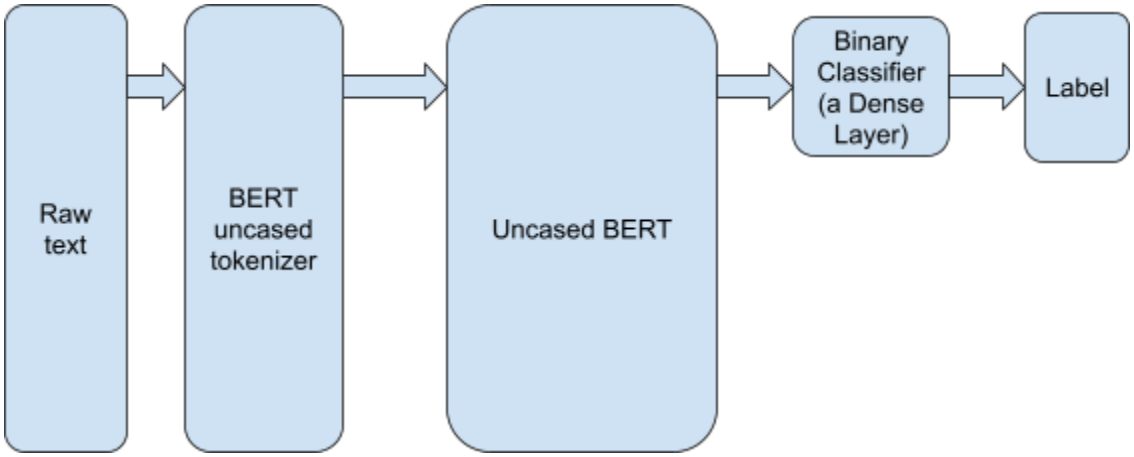
Input: Tokens generated by BERT Tokenizer.

Output: Binary class label (0 or 1) showing whether the tokens are appropriate for bilingual children.

Description: The binary classifier determines whether the tokens contain parts that are not proper for bilingual children. It assigns a binary label (0 or 1) to each input token sequence.

Machine Learning methods

BERT-based Binary Classification Model



Selected Tokenization Method: Uncased BERT Tokenization

We tokenize the raw text into the uncased BERT tokenizer to get the uncased tokenized text.

Uncased BERT Tokenizer VS Cased BERT Tokenizer

Cased BERT Tokenizer could be better in some cases where cased words are essential. For example, for Named-Entity recognition and Part of Speech tagging, people mostly use cased BERT Tokenizer. However, in our application, we only focus on the appropriateness and readability of the sentences. The fact of cased or uncased does not affect any of the criteria. Hence, we decided to use an uncased BERT tokenizer.

Selected Classification Method: Uncased BERT-based Binary Classifier

We feed the tokenized text into the pre-trained uncased BERT model. Notice that the parameters of the BERT model are frozen, we only use BERT to get a representation of the whole text.

Finally, we feed the output of BERT into a binary classifier to predict the binary label of the input text. The parameters of the classifier are fine-tuned to make the model fit with the downstream classification task.

Notice that there is not an activation function. This Classifier is only formulated by a single dense layer, which does linear operations. Moreover, since it is a binary classification task, we used cross-entropy loss as the loss function.

Justification and comparison with RNN

Recurrent Neural Network could achieve better performance than Bert-based models with a small dataset(Ezen-Can,2020). However, the definition of 'small' here is around 15000 samples. It does not match with our case. Our final dataset is extremely small and it only has around 1400 data samples, which is unexpected in our proposal.

Using the pre-trained BERT, we could capture general language patterns and the underlying relationships between words while having a limited dataset. Hence, we decided to use the BERT-based model instead of RNN.

Experiments and results

Hardware and software environment set up

This project was implemented on Google Colab using the following specifications:

Hardware: GPU T4, GPU RAM: 15GB, System RAM: 51GB, Disk: 200GB

Training time: 6 hours

Software: Python 3 Google Colab Engine

Libraries used: Pandas, NumPy, TensorFlow, PyTorch, Hugging Face, Chamd

Parameter and optimizer settings

We used the following hyperparameters to form a total of 32 hyperparameter sets.

Batch size	16		32	
Number of epochs	3	5	7	9
Learning rate	1e-5	2e-5	3e-5	5e-5

Additionally, an AdamW optimizer is applied in the model. AdamW decouples weight decay from the optimization process. This means that in AdamW, the learning rate and weight decay are optimized separately. Thus, AdamW usually gets better training accuracy and makes the model generalize better than using Adam optimizer(Graetz, 2020). Hence, we chose to use the AdamW optimizer.

Metrics

For training, we used accuracy to evaluate the model. For testing, we used accuracy and F1-score to evaluate the model.

Accuracy indicates how often the classification model is correct overall while F1-score integrates precision and recall into a single metric, focusing on the quality of positive and negative predictions at the same time.

Discussion on results

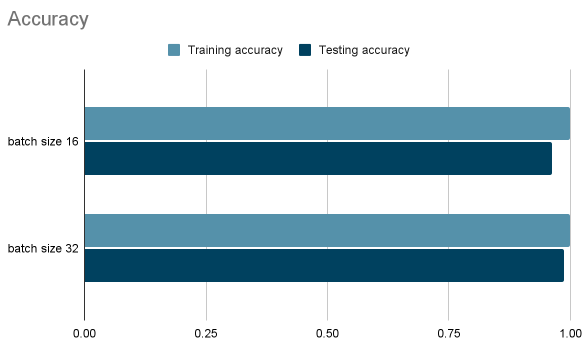
The effect of batch size

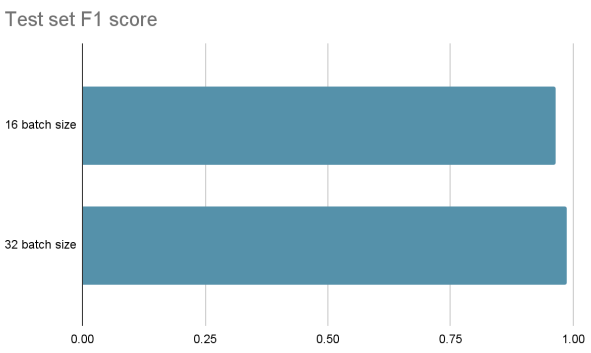
Assuming the hyperparameters excluding the batch size are constant,

Set: number of epochs = 9 , number of learning rate= 5e-5

Training set	
Batch size	accuracy
16	0.9991
32	1

Test set		
Batch size	accuracy	F1-score
16	0.9618	0.9635
32	0.9861	0.9869



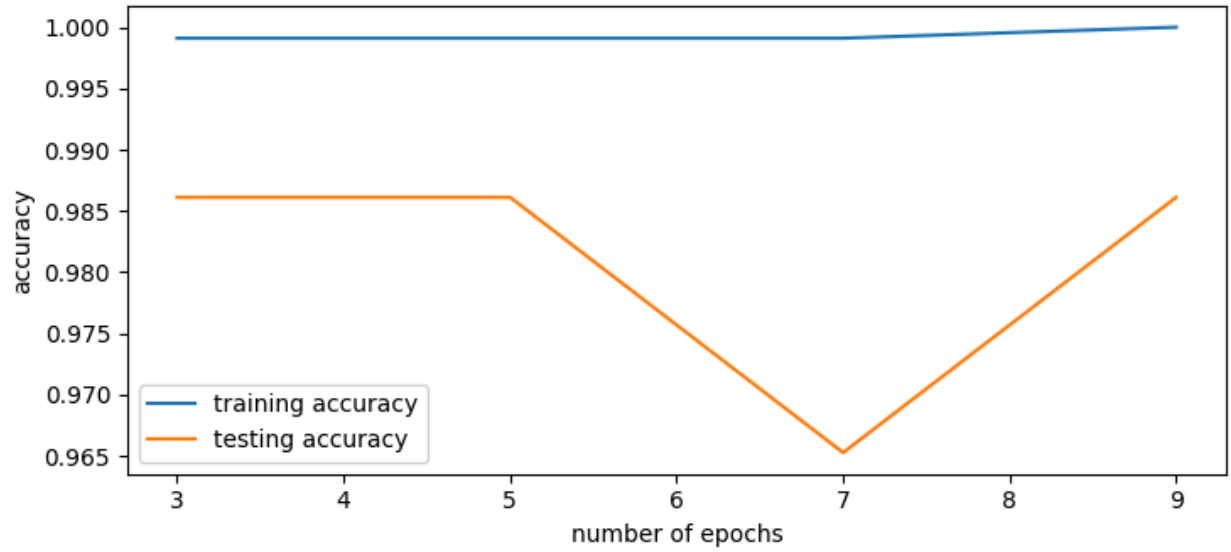


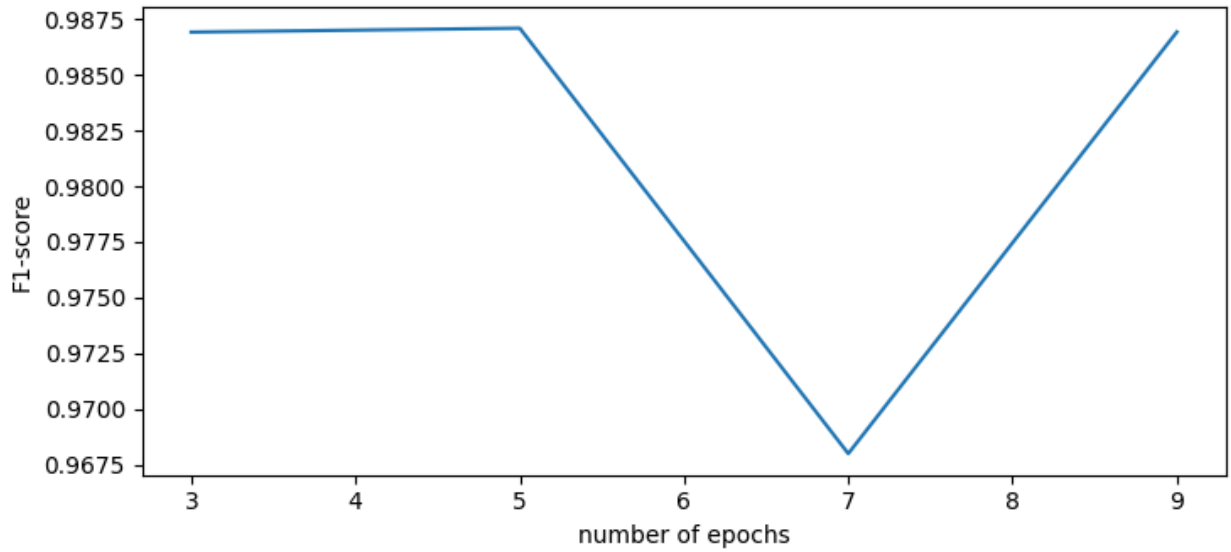
From the above, we could conclude that 32 batch size is better than 16 batch size as the accuracy on both sets with 32 batch size is better than that with 16 batch size. Moreover, the F1-score of 32 batch size is better than that of 16 batch size.

The effect of the number of epochs

Assuming the hyperparameters excluding the number of epochs are constant,

Set: batch size=32 , learning rate= 5e-5



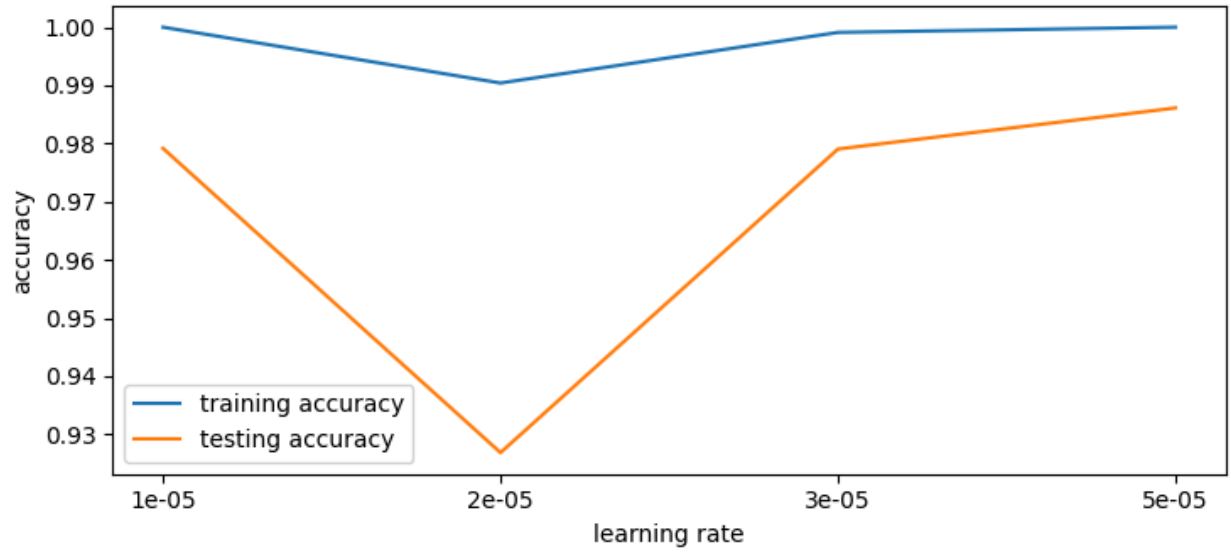


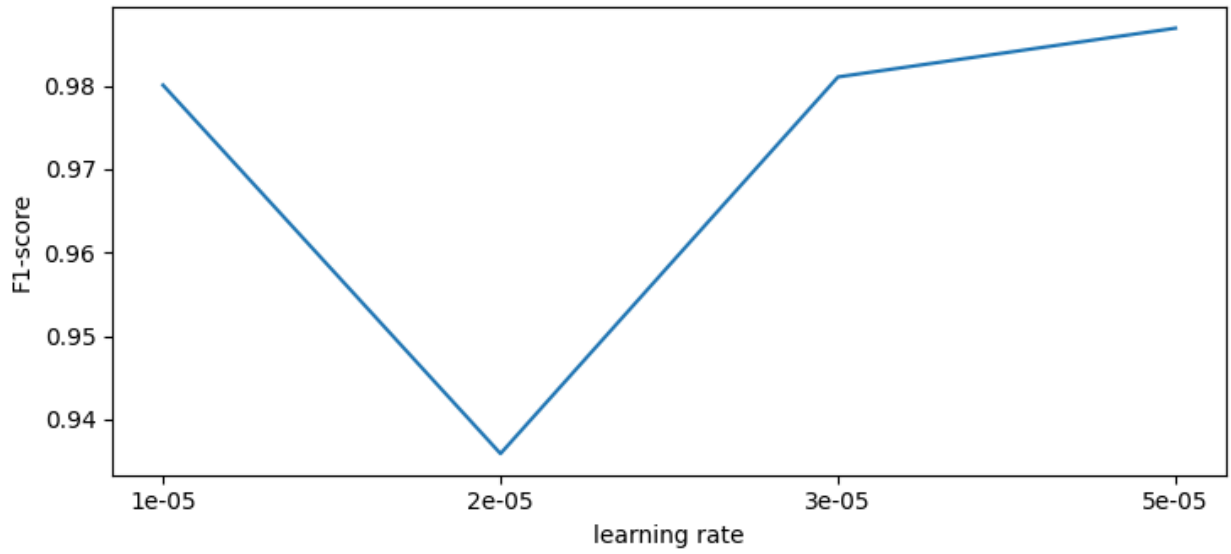
The graph shows that both metrics don't change much for different numbers of epochs except for 7 epochs, where we see a significant sudden drop in testing accuracy and F1 score.

The effect of learning rate

Assuming the hyperparameters excluding the learning rate are constant,

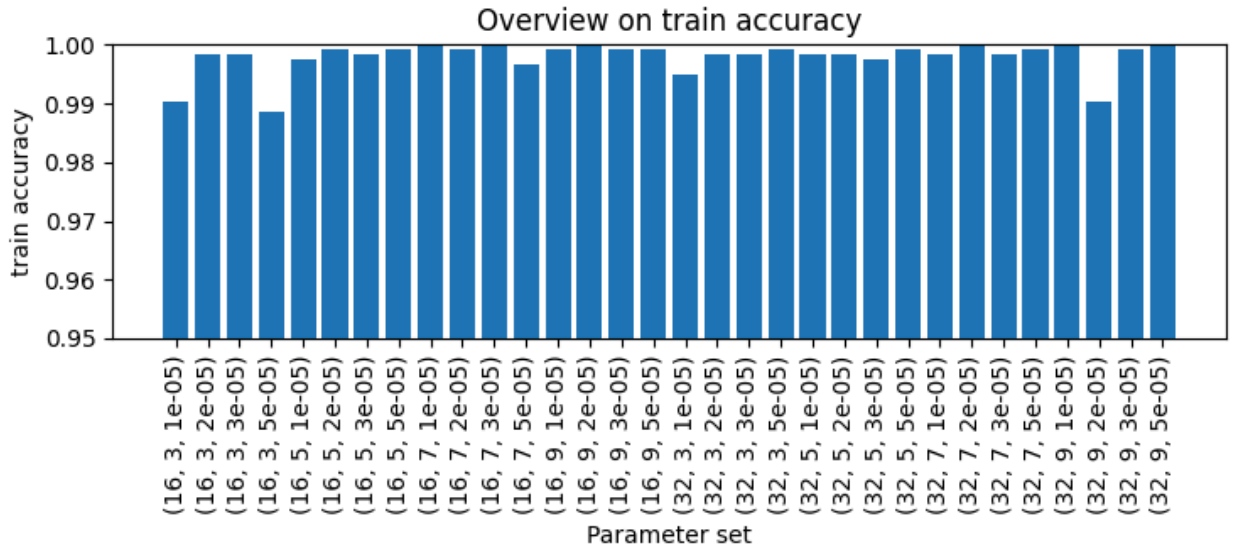
Set: batch size=32 , number of epochs=9

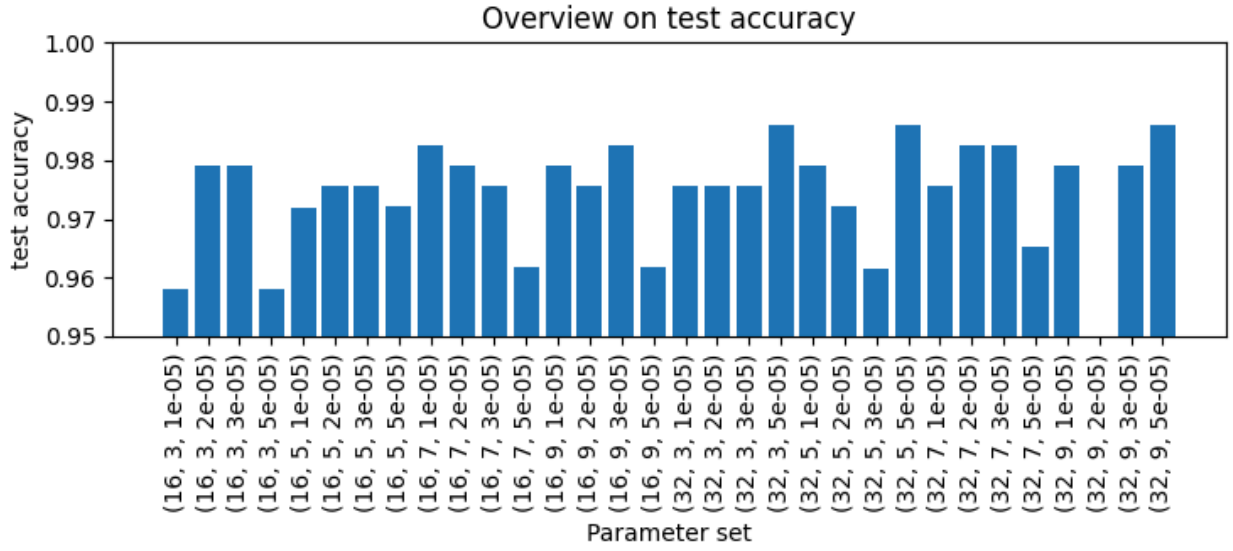


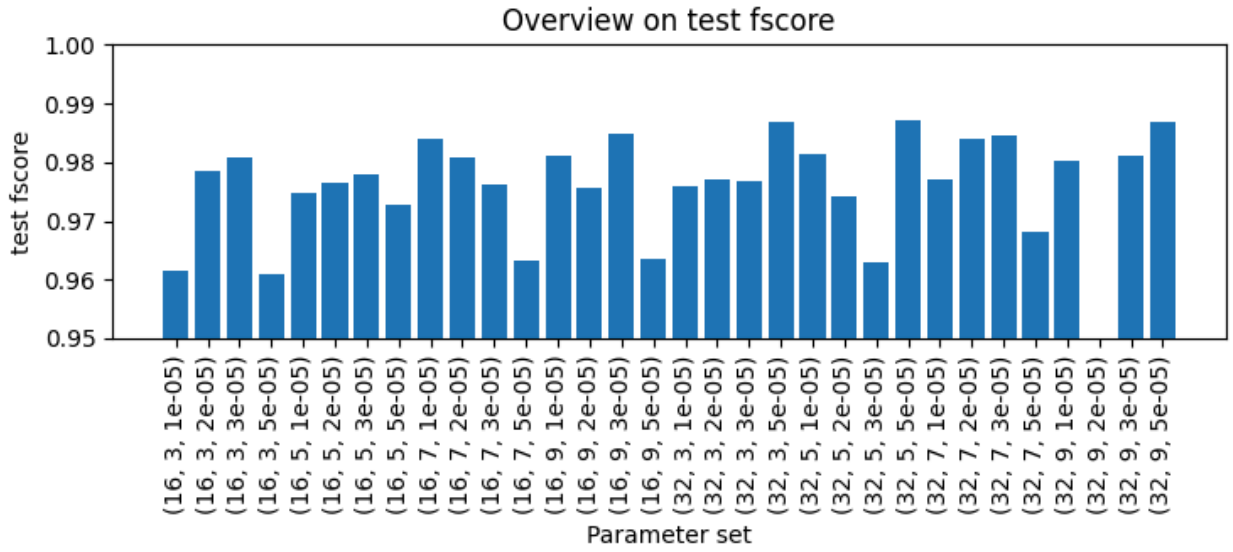


From the graph, we can clearly see that increasing learning rate improves both accuracy and F1 score. Thus, we can conclude that the learning rate of 5e-5 has the best result overall, while 2e-5 has the worst result.

Overview on results of 32 hyperparameter sets







Selected hyperparameter set

Batch-size	Number of epochs	Learning rate
32	9	5e-05

Training accuracy:1 Testing accuracy: 0.9861 Testing F1-score:0.9869

This hyperparameter set has the highest testing accuracy and F1-score among all hyperparameter sets.

Future Improvements

Since the bilingual children's speech corpus only has around 3000 vocabulary size, the calculation of the readability score obviously has a large room for improvement. In the future, if we collect more data in the bilingual children's speech corpus that fills up different vocabulary domains, the prediction could be more comprehensive.

Moreover, the size of the final dataset is too small that we have only more than 1400 samples in our dataset. If we have more than 10000 samples in the dataset, and the

dataset remains balanced, we could achieve a more stable result. In the other hand, we could consider using RNN as well, since research demonstrates that RNN classification model works better than BERT-based classification model with a 15000-entry dataset(Ezen-Can,2020).

Additionally, the accuracy of the model could still be improved. There is room for exploring the hyperparameter fine-tuning as the graph demonstrating the effect of the number of epochs shows that there is potential for the accuracy and F1-score to rise if we increase the number of epochs to a number larger than 9.

Generally, the performance of the classification model is acceptable and good. With the selected parameter set, it could achieve a test set accuracy and F1-score of more than 0.98.

Reference

張雅婷, & 杜潔心. (2016, August 13). 童書扭曲價值觀 說謊「出術」讚機智 [Review of 童書扭曲價值觀 說謊「出術」讚機智]. Hket. https://paper.hket.com/article/1482351/%E7%AB%A5%E6%9B%B8%E6%89%AD%E6%9B%B2%E5%83%B9%E5%80%BC%E8%A7%80%20%E8%AA%AA%E8%AC%8A%E3%80%8C%E5%87%BA%E8%A1%93%E3%80%8D%E8%AE%9A%E6%A9%9F%E6%99%BA

白毅鵬. (2020, 17 June).“粗制濫造如何堂而皇之進了童書--教育--人民網.” Edu.people.com.cn, edu.people.com.cn/BIG5/n1/2020/0617/c1053-31749531.html. Accessed 12 Apr. 2024.

SCHMITT, N., JIANG, X. and GRABE, W. (2011), The Percentage of Words Known in a Text and Reading Comprehension. The Modern Language Journal, 95: 26-43. https://doi.org/10.1111/j.1540-4781.2011.01146.x

google-bert/bert-base-uncased · Hugging Face. (2001, March 11). https://huggingface.co/google-bert/bert-base-uncased

Graetz, F. M. (2020, February 12). *Why AdamW matters - Towards Data Science*. Medium. https://towardsdatascience.com/why-adamw-matters-736223f31b5d

Ezen-Can, Aysu. (2020). A Comparison of LSTM and BERT for Small Corpus.