DS-GA 1011 NLP Assignment 1
Bag of N-Gram Document Classification

Zhengyuan Ding(zd415)

October 10, 2018

1 lntroduction

This report presents the process of hyper-parameters tunning and an analysis of the results, mainly in part 2. In conclusion part, the best model after tunning is compared with the initial model based on the score of the validation accuracy. Finally, the report shows the test result of the final tuned model on the test set.

2 Hyperparameters Tunning

The tokenization scheme set initially is removing punctuations and lowercase. All the hyperparameters that are tuned in this report are shown in Table 1

Variable	Meaning	Tunning range	Initial
`num_epochs`	number of epochs	`num_epochs` ≤ 10	10
`ngram_n`	ngrams	n $\in \{1, 2, 3, 4\}$	1
`max_vocab_size`	maximum vocabulary size	$\{500, 1000, 5000, 10000\}$	1000
`emb_dim`	embedding dimensions	$\{50, 100, 500, 1000]\}$	100
optimizer	optimizer used in the model	Adam/SGD	Adam
`learning_rate`	learning rate	[0.0001, 0.001, 0.01, 0.1] or annealing	0.01

Table 1: Hyperparameters.

2.1 Initial Hyperparameters

A model run with the initial hyper-parameters shows a trend of over fitting (shown in Figure 1). Therefore, to prevent over fitting, we reduce the number of epochs as 6.

2.2 N Grams

Comparing the validation accuracy of using 1gram, 2gram, 3gram or 4gram with other hyperparameters unchanged, Figure 2 shows that the performance of 1gram is far better than the one of other three. Thus we choose 1gram in the following tunning process.

2.3 Vocabulary size

Figure 3 shows that the performance of 10000 and 5000 vocabulary size are better than 1000 and 500. 10000 vocab size has slightly higher validation accuracy, thus we choose the 10000 size.

2.4 Embedding size

We tune the embedding size in the range of [50,100,500,1000] with other parameters unchanged. Figure 4 compares the validation accuracy gained from different embedding size. 50 and 100 are close, but we choose 100 embedding size as its accuracy score rises to a max quicker.

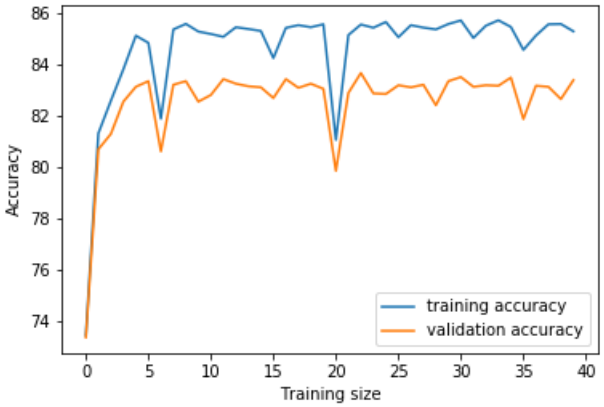

Figure 1: The training and validation accuracy for the initial hyperparameters with 10 epoch. The gap between the training and validation curve shows it's likely to be an overfitting .

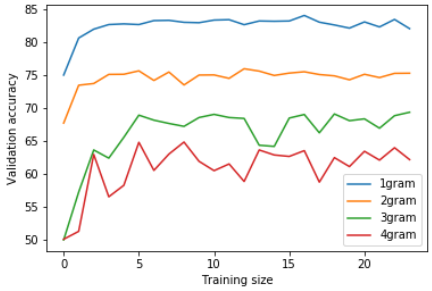

Figure 2: Validation accuracy of using ngrams

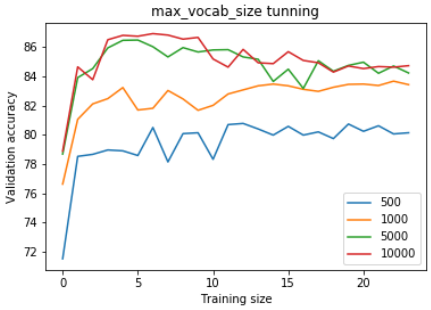

Figure 3: The validation accuracy of using different vocab size

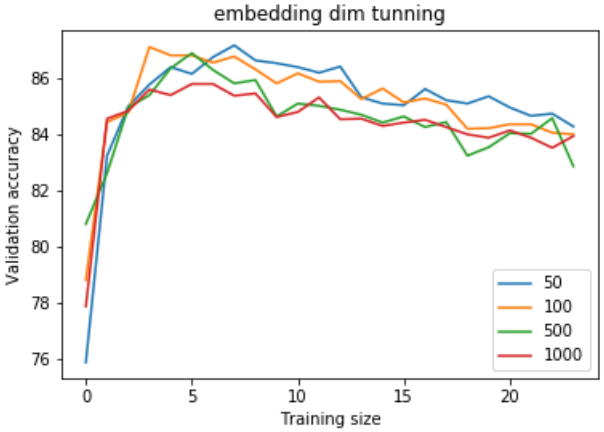

Figure 4: The validation accuracy of using embedding size of [50,100,500,1000]

2.5 Optimizer

With other parameters unchanged, the performance of two optimizers(Adam and SGD) is shown in Figure 5 and Figure 6. It's clear that Adam has a better performance either in training loss and validation accuracy.

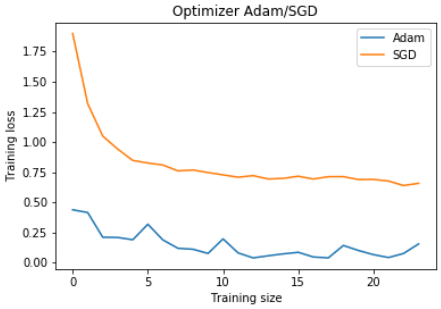

Figure 5: Training loss of two optimizers

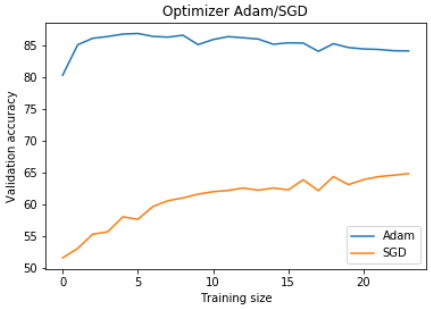

Figure 6: Validation accuracy of two optimizers

2.6 Learning rate

1. Static learning rate range in [0.0001, 0.001, 0.01, 0.1]
   We can see the performance of 0.01 and 0.001 are close in Figure 7 and Figure 8. However, 0.01 learning rate reaches to its highest validation accuracy sooner and have a relatively lower training loss.

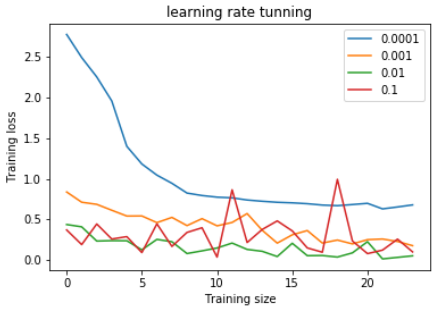

Figure 7: Training loss of different learning rate

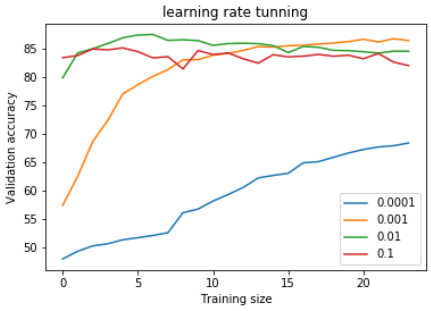

Figure 8: Validation accuracy of different learning rate

2. Static vs Annealing learning rate
   Here we compare the performance of static and annealing learning rate. In particular, we choose linearly and exponentially decay lr formular as:

$$linear : 0.01 - lr \cdot epoch_t$$

$$exponential : 0.01/2^{epoch_t}$$

   The performance is show in Figure 9, where exponentially annealing learning rate reaches the highest validation accuracy. In terms of training loss, the score of three learning rates is close and all of three are vibrating. Therefore, we choose the exponential annealing learning rate and the validation accuracy rises around 87.44 in 6 epochs

2.7 Early Stopping

After tunning all the hyperparameters we add an early stopping to further prevent overfitting and speed up the training process. We check the validation accuracy every 4 batches and set the patience to be 2 periods. If the accuracy doesn't increase larger than 0.01 in 2 periods, then we stop the training process. After adding the early stopping, the model can reach a validation accuracy of 87.14 in 2 epochs.

Figure 9: Training loss of static/annealing learning rate

Figure 10: Validation accuracy of static/annealing learning rate

2.8 Tokenization scheme

The tokenization scheme we use up until now involves removing punctuations and lowercase. Now we experiment more with removing stopwords. The validation accuracy rises to 87.8. Figure 11 and Figure 12

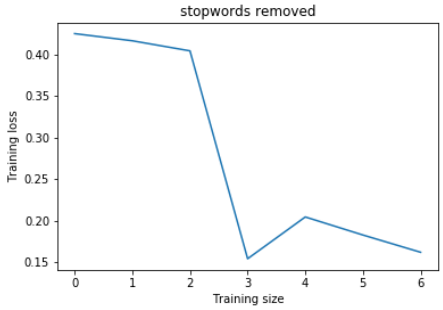

Figure 11: Training loss after stopwords removed

Figure 12: Validation accuracy after stopwords removed

3 Conclusion

After tunning all the hyperparameters, we can reach a validation accuracy of 87.8 and test accuracy of 86.648. The performance comparison between final and initial model is shown in the Table 2.

Variable	Meaning	Final	Initial
`num_epochs`	number of epochs	6	10
`ngram_n`	ngrams	1	1
`max_vocab_size`	maximum vocabulary size	10000	1000
`emb_dim`	embedding dimensions	100	100
optimizer	optimizer used in the model	Adam	Adam
`learning_rate`	learning rate	exponentially annealing from 0.01	0.01
	tokenization scheme	punct,stopword,lowercase	punct,lowercase
		`val_acc` = 87.8	`val_acc` = 83.42

Table 2: Comparison of final and initial models.

Link for code:
https://github.com/CarolD413/DS1011-NLP-2018Fall/blob/master/Assignment1.ipynb