Acad. Year

2025-03

IS6751 Text & Web Mining

Take Home Project 1

LIANG HOU - G2401631D

Wee Kim Wee School of Communication and Information

Nanyang Technological University, Singapore

TASK 1
Build a Sentiment Classifier
with the given requirements.

Wee Kim Wee School of Communication and Information

SUBMITTED BY

LIANG HOU

(G2401631D)

1 Task 1 Introduction

This report presents a sentiment classification task using multilayer neural networks (MLP) based on the dataset `reviews_with_splits_lite.csv`. The baseline model is derived from `3_5_Classifying_Yelp_Review_Sentiment.ipynb`. Various approaches, including text preprocessing and hyperparameter tuning, were applied to improve performance. The report documents the test results of different configurations and discusses the impact of these changes.

2 Baseline Model

The initial model was trained using a vocabulary size of 2256. The test results were as follows:

- Test Loss: 0.345
- Test Accuracy: 85.64%

3 Experiment 1.1: Text Pre-processing

Initially, we attempted several text preprocessing techniques, including case folding, removing special characters, and lemmatization. However, these modifications unexpectedly resulted in a drop in test accuracy from **85.64**% to **83.50**%.

A possible reason for this decline is that removing special characters might have eliminated meaningful punctuation that contributes to sentiment (e.g., exclamation marks indicating strong emotions). Additionally, lemmatization may have altered words in a way that reduced their discriminative power for sentiment classification. For example, converting words like ”amazing” to ”amaze” might strip away sentiment intensity.

Another factor could be that the original vocabulary was well-optimized for the dataset. Lowercasing all text and modifying word forms could have reduced the effectiveness of the learned word embeddings. Since neural networks rely heavily on data distribution, these preprocessing steps might have disrupted useful word patterns captured by the model.

Given this negative impact, we chose to abandon extensive preprocessing and focus on hyperparameter tuning instead, which led to improved accuracy. This result suggests that in sentiment classification tasks, preserving the original text structure may sometimes be more beneficial than aggressively normalizing it.

4 Experiment 1.2: Hyperparameter Tuning

Given the ineffective pre-processing changes, efforts were directed toward hyperparameter tuning.

4.1 Effect of Frequency Cutoff

The frequency cutoff controls the minimum occurrence of words included in the vocabulary. The results are summarized in Table 1.

Frequency Cutoff	Test Loss	Test Accuracy (%)
25	0.390	83.50
15	0.330	86.13
10	0.330	86.72
5	0.309	87.60
3	0.313	87.60
2	0.303	87.50
0	0.307	87.40

Table 1: Test Results for Different Frequency Cutoff Values

A frequency cutoff of 3 was chosen for further tuning.

4.2 Effect of Hidden Layer Dimension

Next, the number of hidden units was adjusted while keeping frequency cutoff at 3. Table 2 shows the results.

A hidden dimension of 20 was selected for further tuning.

Hidden Dimension	Test Loss	Test Accuracy (%)
20	0.313	87.60
25	0.312	87.50
50	0.318	87.40

Table 2: Test Results for Different Hidden Layer Dimensions

4.3 Effect of Batch Size

Batch size was varied while keeping the frequency cutoff at 3 and hidden dimension at 20. Table 3 summarizes the results.

Batch Size	Test Loss	Test Accuracy (%)
64	0.359	87.04
128	0.313	87.60
256	0.311	87.79
512	0.315	87.60

Table 3: Test Results for Different Batch Sizes

A batch size of 256 was chosen.

5 Conclusion

The best performing model was achieved with the hyperparameter settings in the attached *.ipynb* This configuration resulted in the following.

- Test Loss: 0.311
- Test Accuracy: 87.79%

Compared to the baseline model, the best configuration improved test accuracy from 85.64% to 87.79%, demonstrating the effectiveness of hyperparameter tuning over text pre-processing.

TASK 2

Build a Surname Classifier

with the given requirements.

Wee Kim Wee School of Communication and Information

SUBMITTED BY

LIANG HOU

(G2401631D)

6 Task 2 Introduction

This experiment focuses on implementing and optimizing a multilayer perceptron (MLP) model for surname classification. The dataset used is `surnames_with_splits.csv` from the `mlp_surnames` repository, and modifications were made based on hyperparameter tuning and text preprocessing techniques to improve the model's performance.

7 Methodology

7.1 Baseline Model

The baseline model was implemented using the code from `4_2_Classifying_Surnames_with_an_MLP.ipynb`, with an initial test loss of 1.6680 and an accuracy of 48.5%. The model consists of a multilayer neural network without recurrent architectures (e.g., RNN, LSTM, CNN).

7.2 Text Preprocessing and Feature Engineering

To enhance feature representation, we experimented with multi-gram character embeddings by adjusting the `gram` parameter:

Multi-gram	Test Loss	Test Accuracy (%)
1 (Baseline)	1.6680	48.5
2	1.3687	67.3
3	1.6136	64.7
4	1.9327	59.4

Table 4: Impact of multi-gram character representations

Analysis: The results indicate that using 2-gram features significantly improves accuracy (67.3%), but increasing to 3-gram and 4-gram leads to performance degradation. This suggests that higher-order n-grams introduce noise rather than useful patterns.

7.3 Hidden Dimension Tuning

Different values of `hidden_dim` were tested:

Hidden Dim	Test Loss	Test Accuracy (%)
100	1.7242	69.0
200 (Optimal)	1.3206	67.0
300	1.4734	69.8
400	1.3750	64.6

Table 5: Impact of hidden dimension size

Analysis: The best performance was achieved with `hidden_dim` = 200, balancing complexity and generalization. While 300 provided the highest accuracy, its loss was higher, indicating potential overfitting.

7.4 Epoch and Early Stopping

We explored different training durations:

Num Epochs	Test Loss	Test Accuracy (%)
100 (Optimal)	1.4841	70.3
150	1.9944	70.6
200	2.3910	71.6

Table 6: Effect of increasing training epochs

Analysis: Extending training epochs increased accuracy but also raised test loss, indicating overfitting. Thus, `num_epochs` = 100 was chosen to balance accuracy and generalization.

7.5 Early Stopping Criteria and Batch Size

Adjusting early stopping and batch size:

Early Stopping	Test Loss	Test Accuracy (%)
5 (Optimal)	1.4841	70.3
10	2.5389	71.6

Table 7: Effect of early stopping criteria

Analysis: Increasing early stopping criteria improved accuracy slightly but significantly worsened test loss, confirming overfitting.

8 Final Model and Conclusion

Based on these experiments, the final model parameters were:

Parameter	**Value**
gram	2
hidden_dim	200
num_epochs	100
early_stopping_criteria	5
learning_rate	0.001
batch_size	64

Table 8: Model Hyperparameters

The final model achieved:

- **Test Loss**: 1.3206
- **Test Accuracy**: 67.1%

While some configurations achieved higher accuracy, they exhibited overfitting, as indicated by increased loss. The final model was selected for its balance between accuracy and generalization. Future improvements could involve additional regularization techniques, data augmentation, or alternative activation functions.