Phuoc Le, Gil Rabara, Dilnoza Saidova, Vivian Tran
TCSS 456: Natural Language Processing
May 28, 2023

Deliverable 2 - Fine Tuning

Source Code Files

Source code is written and developed in the Google Colab; everything is uploaded to the group project folder as a zip file. No changes were made to the original code other than an increased number of epochs in the model training stage.

Changes to Input Representation

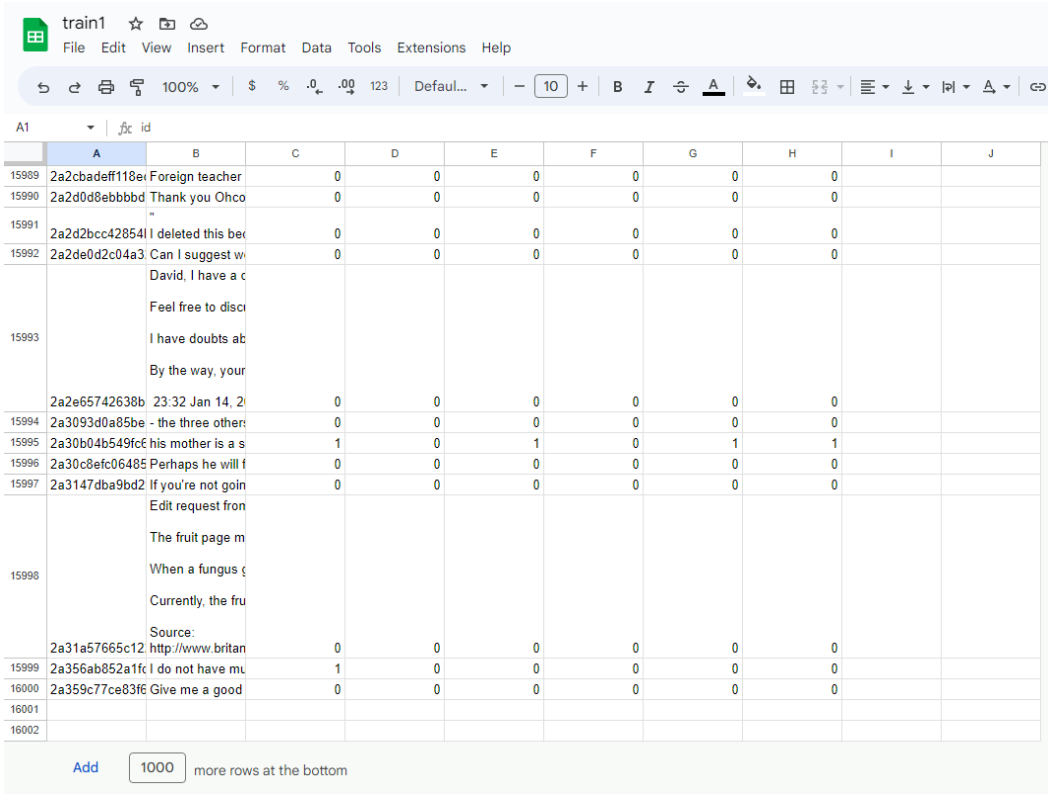
The original input representation was kept and implemented. No changes were made.

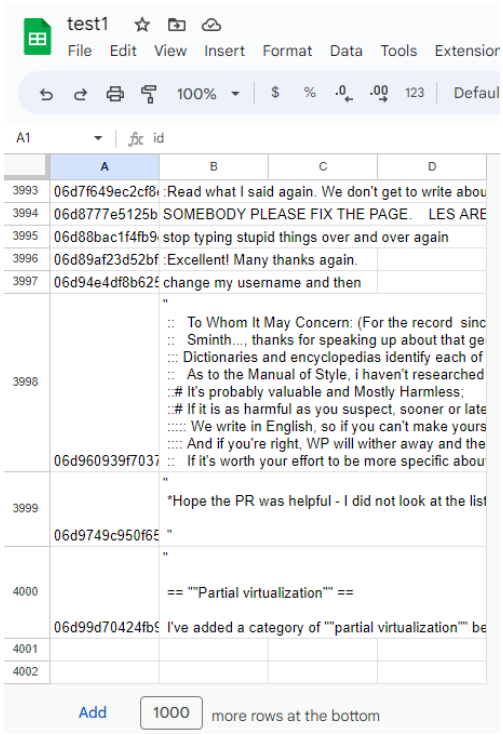
Implementation of a Different Model

The original model was kept and implemented. No changes were made.

Changes to the Training and Testing Process

While no changes to the training and testing process were made, slight modifications to the training and testing CSV files were made. Specifically, the number of training instances was increased to 16,000 and testing instances were decreased to 4,000. The original training and testing data set size was 10,000 for both instances. This way, the train-to-test ratio is 80:20.





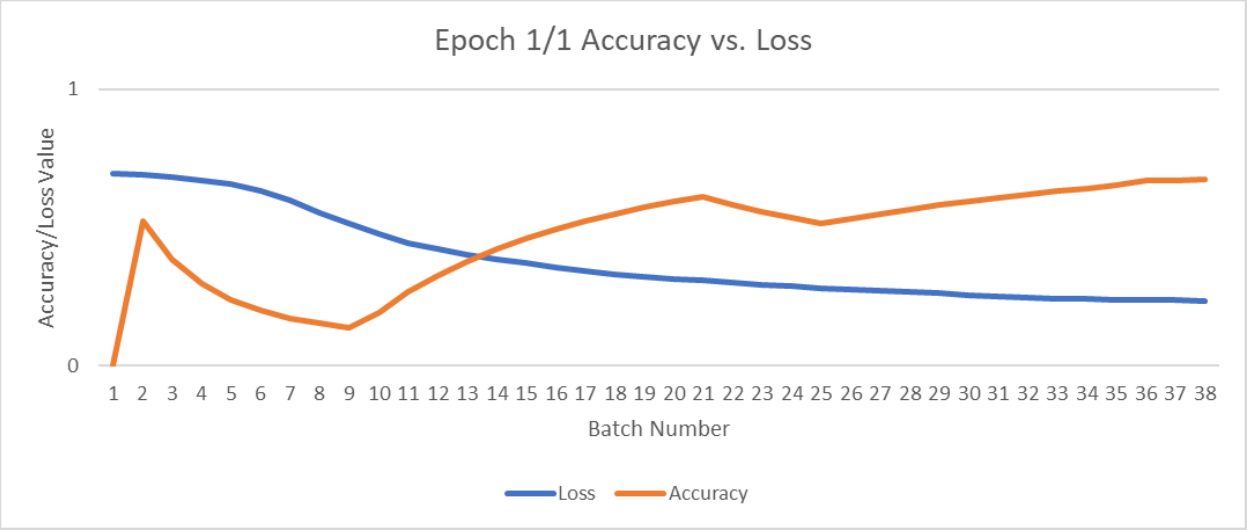
Evaluation Results

Epoch 1/1 Accuracy vs. Loss Results

The highest accuracy of the model with one epoch was 0.6747.

Batch Number	Loss	Accuracy
1	0.6977	0.0039
2	0.6901	0.5234
3	0.6821	0.3841
4	0.6715	0.2959
5	0.656	0.2391
6	0.6334	0.2012
7	0.6001	0.1735
8	0.5525	0.1528
9	0.514	0.1363
10	0.4775	0.1941
11	0.4448	0.267
12	0.4225	0.3271
13	0.4024	0.3786
14	0.3863	0.4224
15	0.3709	0.4607
16	0.3553	0.4941
17	0.344	0.5237
18	0.3321	0.549
19	0.3225	0.5726
20	0.3145	0.5934
21	0.308	0.6122
22	0.2998	0.5843
23	0.2936	0.5589
24	0.2871	0.5358

25	**0.28**	**0.5144**
26	**0.2752**	**0.5329**
27	**0.2703**	**0.5501**
28	**0.2672**	**0.566**
29	**0.262**	**0.5808**
30	**0.2565**	**0.5948**
31	**0.2513**	**0.6077**
32	**0.248**	**0.6198**
33	**0.2449**	**0.631**
34	**0.2431**	**0.6418**
35	**0.2407**	**0.652**
36	**0.2368**	**0.6703**
37	**0.2368**	**0.6703**
38	**0.2356**	**0.6747**



Model Optimization - Two Epochs

As the number of epochs increased to two, so did the accuracy of the model. On the first epoch, the highest accuracy was 0.6301. On the second epoch, the accuracy drastically increased to over 0.99, with higher accuracy of 0.9919.

Some of the reasons for this were improved model optimization, generalization, regularization, and learning complex patterns:

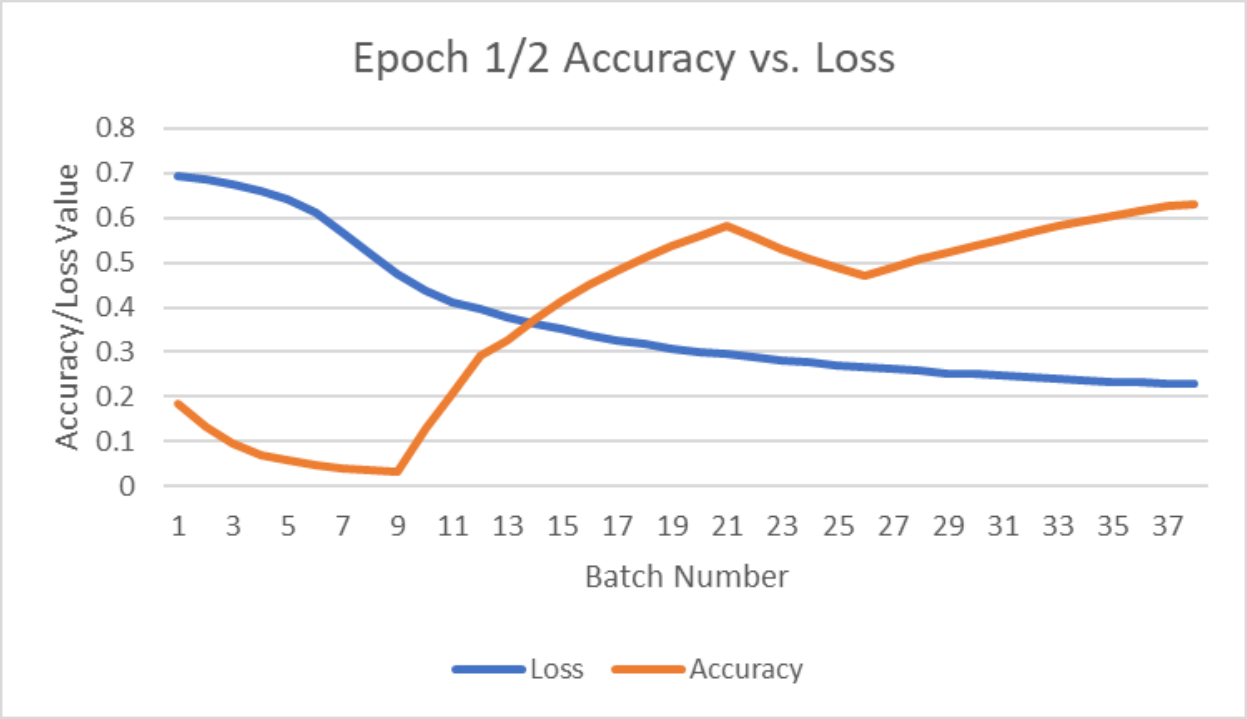
- **Model Optimization**
  - Allowing the model to go through more epochs gives it more opportunities to fine-tune its parameters and improve its performance. With each epoch, the model learns to make better predictions, leading to increased accuracy.
  - During training, the model's parameters (weights and biases) are updated based on the optimization algorithm (e.g., gradient descent) and the error between the predicted outputs and the true labels.
- **Generalization**
  - As the number of epochs increases, the model learns to generalize better and capture the underlying patterns, leading to improved accuracy on unseen data.
  - Initially, the model may focus on learning specific details and noise present in the training data, resulting in poorer performance on the validation or test data.
- **Regularization**
  - By training the model for more epochs, the regularization techniques (i.e., dropout used to prevent overfitting) have more time to effectively regularize the model, reducing overfitting and improving generalization, which can result in increased accuracy.

Epoch 1/2 Accuracy vs. Loss Results

Batch Number	Loss	Accuracy
1	**0.694**	**0.1836**
2	**0.6863**	**0.1309**
3	**0.6764**	**0.0938**
4	**0.662**	**0.0703**
5	**0.6418**	**0.057**
6	**0.6124**	**0.0482**
7	**0.569**	**0.0413**
8	**0.5186**	**0.0361**
9	**0.4756**	**0.0321**

10	0.4377	0.1285
11	0.4122	0.207
12	0.3948	0.2928
13	0.3777	0.3275
14	0.3639	0.375
15	0.3505	0.4159
16	0.3376	0.4519
17	0.3264	0.4835
18	0.3195	0.5119
19	0.309	0.5374
20	0.3014	0.5604
21	0.2948	0.5809
22	0.2876	0.5545
23	0.2816	0.5304
24	0.2756	0.5085
25	0.2712	0.4881
26	0.2667	0.4695
27	0.2626	0.4887
28	0.2576	0.5067
29	0.2529	0.5234
30	0.2504	0.5392
31	0.2466	0.5537
32	0.2434	0.5676
33	0.239	0.5807
34	0.236	0.5927

35	0.2334	0.6041
36	0.2319	0.6149
37	0.2303	0.6253
38	0.2298	0.6301

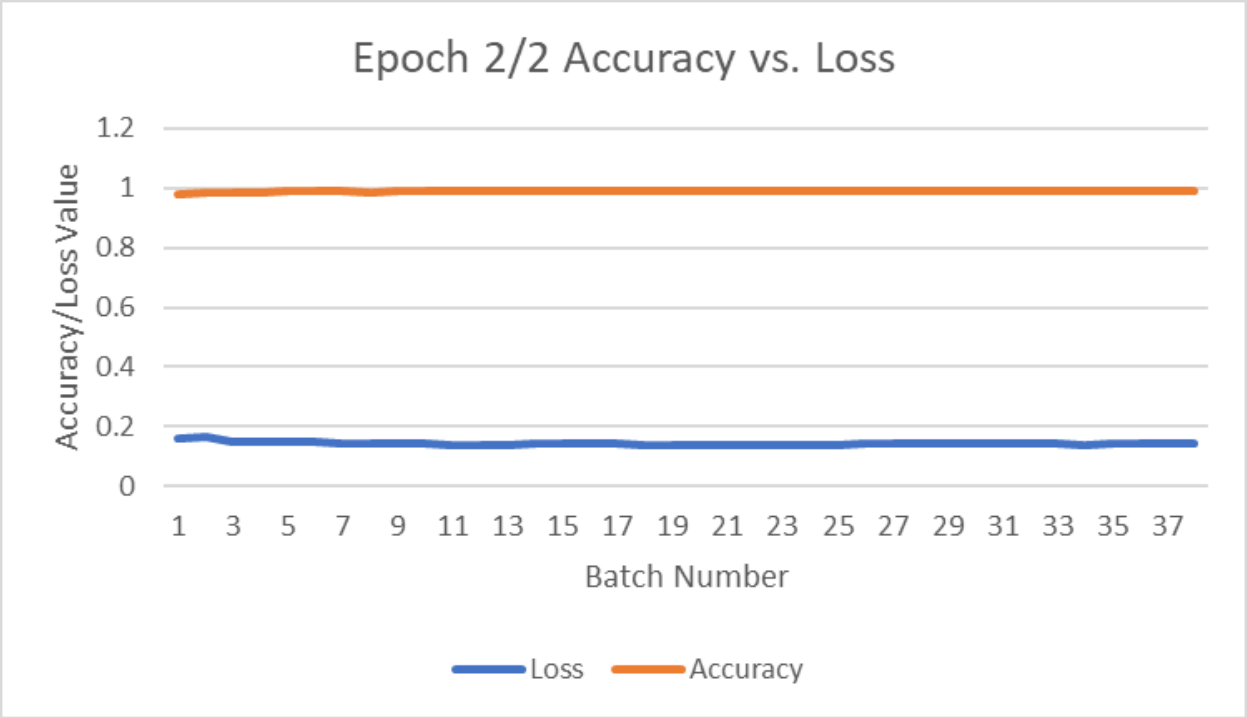


Epoch 2/2 Accuracy vs. Loss Results

Batch Number	Loss	Accuracy
1	0.1579	0.9766
2	0.1636	0.9824
3	0.1486	0.9844
4	0.146	0.9873
5	0.1496	0.9883
6	0.1466	0.9889
7	0.1448	0.99
8	0.1419	0.9863

9	0.1408	0.9878
10	0.1422	0.9887
11	0.139	0.989
12	0.1388	0.9896
13	0.1382	0.9895
14	0.1429	0.9894
15	0.1404	0.9898
16	0.144	0.9905
17	0.1404	0.991
18	0.139	0.9913
19	0.138	0.9916
20	0.1377	0.9918
21	0.139	0.9918
22	0.1381	0.9917
23	0.139	0.9918
24	0.1388	0.9917
25	0.1396	0.992
26	0.1409	0.9919
27	0.1413	0.9919
28	0.1414	0.9918
29	0.1412	0.9916
30	0.1411	0.9914
31	0.1415	0.9914
32	0.1401	0.9913
33	0.1401	0.9914

34	**0.1397**	**0.9912**
35	**0.1401**	**0.9912**
36	**0.1405**	**0.9906**
37	**0.1406**	**0.9905**
38	**0.1403**	**0.9905**



Observations

How well is the task being done?

The model's performance can be assessed by looking at the accuracy values for both the training and validation datasets.

For the training dataset:

- Training loss: 0.2298
- Training accuracy: 0.6301

For the validation dataset:

- Validation loss: 0.1436
- Validation accuracy: 0.9948

The accuracy metric represents the proportion of correctly classified instances compared to the total number of instances in the dataset. A higher accuracy value indicates that the model is making more accurate predictions.

In this case, the model achieves a training accuracy of 0.6301, which suggests that it performs reasonably well on the training data. However, it is important to note that the training accuracy alone may not provide a complete picture of the model's performance, as it could still be overfitting to the training data.

The validation accuracy is a more critical measure of the model's performance. With a validation accuracy of 0.9948, the model performs exceptionally well on the validation dataset. This high accuracy suggests that the model is generalizing effectively to unseen data and has learned relevant patterns and features.

Is the model overfitting or underfitting?

The model achieves relatively high accuracy on the training dataset (0.6301), indicating that it performs reasonably well on the data it was trained on. However, it's essential to consider the performance of the validation dataset to assess whether the model is overfitting or underfitting.

The validation accuracy is also quite high (0.9948), indicating that the model generalizes well to unseen data. A high validation accuracy suggests that the model is not overfitting to the training data but is able to accurately classify instances from the validation dataset.

Since both the training and validation accuracies are relatively high and have a similar magnitude, and the validation accuracy is not significantly lower than the training accuracy, it is likely that the model is not overfitting. These results suggest that the model is able to generalize well to unseen data and perform well on the task at hand.

Are there any further changes that need to be made in future iterations?

General ideas to be considered for further iterations:

- **Hyperparameter Tuning**
  - Explore different hyperparameter settings to find the optimal combination, including learning rate, batch size, dropout rate, number of LSTM units, kernel size in Conv1D, etc. Hyperparameter tuning techniques like grid search or random search can be employed to systematically search the hyperparameter space.
- **Model Evaluation**
  - Evaluate the model on additional evaluation metrics to better understand its performance. This includes precision, recall, F1 score, or domain-specific metrics that are relevant to the specific task at hand.