UNIVERSITY OF ATHENS
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

Deep Learning for NLP

Student name: *Aikaterini-Methodia Zacharioudaki*

Course: *Artificial Intelligence II (M138, M226, M262, M325)*
Semester: *Spring Semester 2025*

Contents

1. Abstract

The task of this assignment is to develop a sentiment classifier for a tweet dataset. We will use two versions of BERT, BertForSequenceClassification and DistilBertForSequenceClassification, for classification and their corresponding tokenizers to convert our data to vectors.

2. Data processing and analysis

2.1. Pre-processing

We follow the steps below for pre-processing:

- Replace tags and links with the words 'user' and 'url'
- Remove HTML tags and emails
- Replace timestamps with the word 'time'
- Replace all excess whitespace characters with a single space

2.2. Vectorization

We will convert our data to BERT Word Embeddings, using the BertTokenizer and the DistilBertTokenizer, for the Bert and the DistilBert models respectively. BERT produces word representations that are dynamically informed by the words around them, thus capturing differences like polysemy and other forms of information that result in more accurate feature representations, which in turn results in better model performance, compared to other vector representations.

3. Algorithms and Experiments

3.1. Collator

We use the DataCollatorWithPadding to dynamically pad the batches. The mean length of a tweet in all the datasets is 20 tokens long, while the longest tweet is 256 tokens in the training set, 154 tokens in the validation set, and 294 tokens in the test data set. Padding all the tweets to 294 is memory consuming and makes the training and evaluation times too long. The answer to that is the collator, which pads every batch separately, according to the longest tweet of the current batch. That way batches with short tweets use less memory and passing them through the model takes less time, thus making the whole training process shorter.

3.2. Training Function

We implement a simple training function, that contrary to the previous assignment doesn't feature an early stopping mechanism. Bert models only need a couple of epochs before they start overfitting, so instead of early stopping or saving the best model of a specific epoch, we will fine tune the number of epochs using the optuna framework.

3.3. Optimization techniques

We will use the Optuna framework to optimize the models. For both models we will experiment with the optimizer and the number of epochs.

3.4. Hyper-parameter tuning

Specifically, we will experiment with the following parameters:

- epochs : The number of epochs for the training function. We choose from: 2, 3, 4
- optimizer : The optimizer to use. We choose from: torch.optim.Adam, torch.optim.AdamW
- learning_rate : The learning rate used in the optimizer. We choose from: 1e-5, 2e-5, 3e-5, 4e-5, 5e-5

3.5. Experiments

We create an Optuna study with the objective to maximize the accuracy in the validation set, using the optuna.samplers.TPESampler to help the study run more efficiently. Since each trial needs more than a few minutes to complete and the resources available are limited, we choose to run 5 trials for the Bert model and 15 trials for the DistilBert model

3.5.1. Table of trials for Bert.

Trial	epochs	optimizer	learning_rate	Score
1	4	AdamW	5e-05	0.8441
2	4	AdamW	4e-05	0.8471
3	3	Adam	2e-05	0.8517
4	2	Adam	4e-05	0.8543
5	2	Adam	2e-05	0.8541

Table 1: Trials for Bert

3.5.2. Table of trials for DistilBert.

Trial	epochs	optimizer	learning_rate	Score
1	4	AdamW	5e-05	0.8354
2	4	AdamW	4e-05	0.8394
3	3	Adam	2e-05	0.8468
4	2	Adam	4e-05	0.8474
5	2	Adam	2e-05	0.8480
6	4	AdamW	5e-05	0.8366
7	2	AdamW	5e-05	0.8471
8	2	AdamW	1e-05	0.8447
9	3	Adam	5e-05	0.8420
10	2	AdamW	2e-05	0.8478
11	2	Adam	1e-05	0.8446
12	3	AdamW	3e-05	0.8455
13	4	AdamW	1e-05	0.8465
14	2	AdamW	3e-05	0.8484
15	2	Adam	2e-05	0.8481

Table 2: Trials for DistilBert

3.6. Evaluation

To evaluate our model, we will use the following metrics:

- **Accuracy** : The proportion of all predictions that are correct. It is calculated as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Confusion matrix** : In our case a 2×2 matrix that displays the number of true positives, true negatives, false positives, and false negatives. This aids in calculating the accuracy and other metrics.

- **Loss Curve** : A plot of loss as a function of the number of training iterations. The loss curve features the training loss and the validation loss. When plotting both, it measures the model's performance on a separate set of data not used during training, helping identify when the model is just memorizing the training data (overfitting) rather than truly understanding the patterns.

- **ROC curve** : Illustrates the performance of a classifier model at varying threshold values. The ROC curve is the plot of the true positive rate (TPR) against the false positive rate (FPR) at each threshold setting.

4. Results and Overall Analysis

4.1. Results Analysis

After two epochs, the Bert model has an accuracy score of 0.8558 and a loss score of 0.3660 in the validation set, while the DistilBert model reaches 0.8478 and 0.3630 respectively. We can see that both models perform very well, a fact strengthened by the the large number of TP and TN in the confusion matrix as well as the expansive area under the ROC Curve. If we study the loss curve, we can see that while the training loss lessens, the validation loss becomes larger, meaning the models slightly overfit. However, we believe it was necessary for the models to train for more than one epoch.

For the test set the Bert model achieves an accuracy of 0.85626, while the DistilBest model achieves an accuracy of 0.84739.

4.1.1. Best trial for Bert.

After running optuna, we find that the best parameters are:

- epochs = 2
- optimizer = Adam
- learning_rate = 4e-05

Metric	Score
Accuracy	0.8558
Recall	0.8520
Precision	0.8586
F1 score	0.8553

Confusion Matrix:

		Predicted	
		0	1
True	0	18223	2974
	1	3136	18063

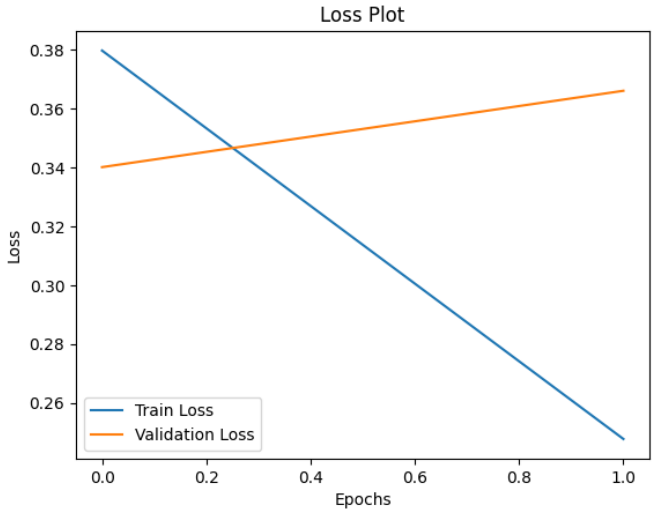

Figure 1: Loss Curve for Bert Model

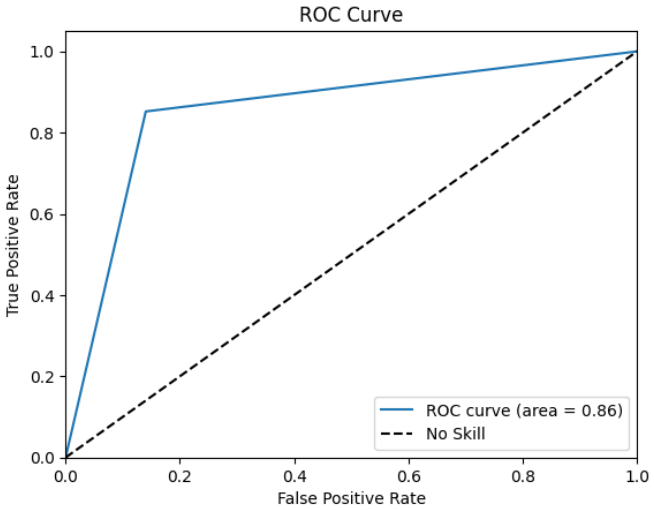

Figure 2: ROC Curve for Bert Model

4.1.2. Best trial for DistilBert.

After running optuna, we find that the best parameters are:

- epochs = 2
- optimizer = AdamW
- learning_rate = 3e-05

Metric	Score
Accuracy	0.8478
Recall	0.8400
Precision	0.8533
F1 score	0.8466

Confusion Matrix:

		Predicted	
		0	1
True	0	18137	3060
	1	3390	17809

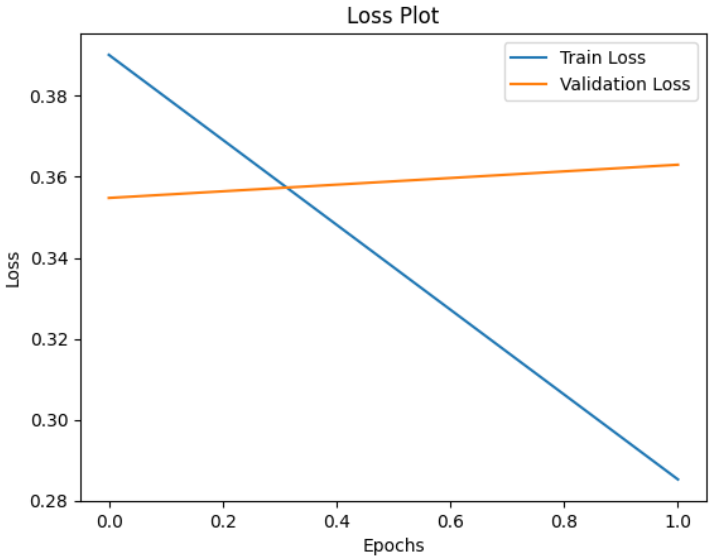

Figure 3: Loss Curve for DistilBert Model

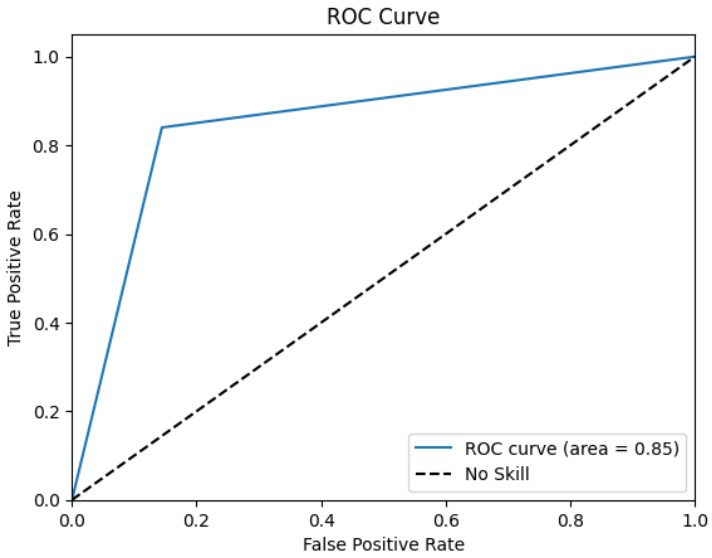

Figure 4: ROC Curve for DistilBert Model