UNIVERSITY OF ATHENS
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

Deep Learning for NLP

Student name: *Chatzispyrou Michail*
sdi: 1115202000212

Course: *Artificial Intelligence II (M138, M226, M262, M325)*
Semester: *Fall Semester 2023*

Contents

1. Abstract

This work studied the development of a sentiment classifier, using Feed Forward Neural Networks, for a twitter dataset regarding the Greek general elections. Each tweet consists of an ID, its actual text and lastly the class it belongs to: POSITIVE, NEUTRAL, NEGATIVE. The classifier handles all the classes. The experiments performed for this development uti- lized a variety of methods, including data processing, word embeddings and hyper-parameter tuning.

2. Data processing and analysis

2.1. Pre-processing

Pre-processing is an essential component of Artificial Intelligence (AI) and Machine Learning (ML), since it has the ability to convert raw input data into a more refined and useful form. Also, it can improve the quality and validity of the input data, which enhances both the efficiency and precision of the algorithms that come after.

An extensive data cleaning procedure was used to improve the dataset's quality and analytical appropriateness in order to create the current model. The act of removing links was crucial in getting rid of unnecessary site addresses and maintaining the dataset's primary emphasis. Similarly, by removing potential distractions from metadata, the systematic elimination of mentions and hashtags improved the dataset's readability. Text capitalization was standardized to lowercase in order to preserve uniformity throughout the dataset. Moreover, a further refinement phase required keeping only Latin or Greek-language words in order to guarantee linguistic consistency and remove non-alphabetic characters. Lemmatization was included as an additional option to the data cleaning process to further simplify the dataset in order to enable a more accurate and comprehensive analysis in the present study. Additionally, the removal of stopwords improved dataset conciseness by excluding common words with limited analytical value. Finally, accents were removed to ensure uniformity in text representation.

In the previous assignment, we investigated different pre-processing methods to improve our models' performance. We looked into three distinct strategies:

- No Pre-Processing (NP), the dataset was intact
- Basic Pre-Processing (BP), the dataset underwent the following modifications:
  - Removing links
  - Removing hashtags
  - Removing mentions
  - Removing stopwords
  - Removing accents
  - Lowercase
  - Removing non Latin or Greek-language words
- Basic Pre-Processing & Lemmatization (PL), all the Basic Pre-Processing with the addition Lemmatization

Following extensive testing, it became clear that, of the three approaches, Basic Pre-Processing (BP) produced the most encouraging outcomes. These results have led us to decide to simplify our strategy for this task. In order to preserve uniformity and enable a more precise comparison amongst models, we will just utilize the fundamental pre-processing method.

2.2. Analysis

We use a variety of tools to navigate the data's complexities as we begin our initial exploration phase and reveal its mysteries. Printing the dataframe (1) itself is the first step that is both straightforward and essential. Having a basic knowledge of the dataset's structure from this first look lets us examine its elements, column names, and the type of data it contains.

	New_ID	Text	Sentiment	Party
0	35027	#απολυμανση_κοριοι #απεντομωση_κοριος #απολυμανσεις #κοριος#Alphatv #Την Κυριακη #Κουλης #Τσιπρ...	NEUTRAL	SYRIZA
1	9531	Έξι νέες επιστολές για τη Μακεδονία «καίνε» τη ΝΔ - Ο Μητσοτάκης γνώριζε και δίχασε το έθνος htt...	NEGATIVE	ND
2	14146	Ισχυρό ΚΚΕ, δύναμη του λαού στη Βουλή και στους καθημερινούς αγώνες	POSITIVE	KKE
3	28716	@five2nds @anthi7vas Μνημονιακότατο το #ΜεΡΑ25 #Εκλογες_2019 #8iouliou #epomeni_mera #ΤΩΡΑ_ΚΚΕ ...	NEUTRAL	KKE
4	32886	@ai_katerina Αυτό που είναι συγκλονιστικό είναι η ψυχασθένεια του Τσίπρα!	NEUTRAL	SYRIZA
...	...	...	...	...
36625	35374	@KourtakisJohn @kmitsotakis Ο Κούλης ο Μητσοτάκης λέει ψέματα!!!Δεν άδειασε κανένα Μπάμπη Παπαδη...	NEUTRAL	ND
36626	7744	@enikos_gr @NChatzinikolaou @AdonisGeorgiadi Πρόσεξε μην σκίσει και κανένα καλσόν. Επίσης ... χά...	NEGATIVE	ND
36627	35216	Η θέση του ΚΚΕ για την ασφάλεια των πολιτών και τους διάφορους Ρουβίκωνες είναι η βαθιά κρίση το...	NEUTRAL	KKE
36628	2855	@thanosplevris Μαρη κακομοίρα θυγατέρα του ναζιστη αντισημίτη, έχεις ξεφτιλιστεί τόσο πολύ που ο...	NEGATIVE	ND
36629	25500	@gijjstalking @SpirosR76 Εντάξει με έπεισες! Και εγω ΚΚΕ! ✌️❤️	NEUTRAL	KKE

36630 rows × 4 columns

Figure 1: Train DataFrame

A snapshot of our dataset, consisting of a significant 36630 rows and carefully arranged into 4 columns, appears as we print the dataframe (1). Every tweet is given a unique identity by the first column. The tweets themselves are visible in the second column, which provides an insight into the textual world. Each tweet is a story contained inside the boundaries of the dataset. Furthermore, in the third column, we find the sentiment classes of neutral, positive, and negative. The last touch is found in the fourth column, where each tweet is matched with a particular political party based on its political resonance.

Next, we turn our attention to visualization, creating a bar plot (2) that provides a broad overview of the political fields included in the dataset. This visual aid offers a brief overview of how tweets are distributed among different political parties.

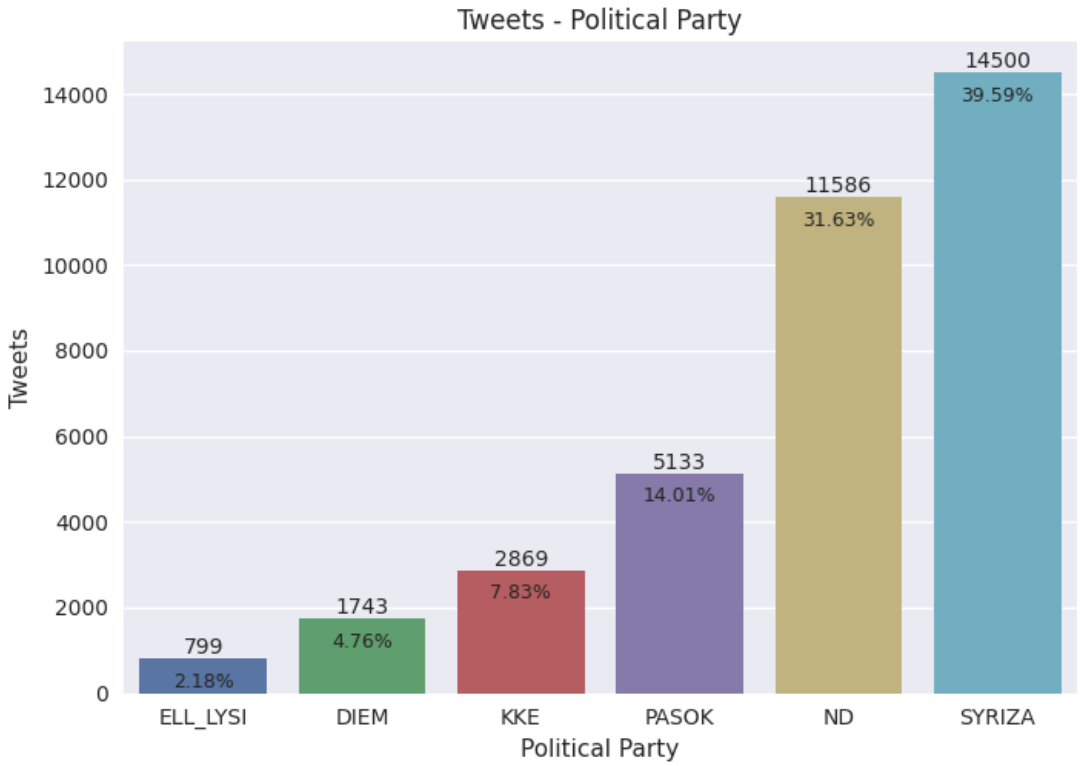

Figure 2: Tweets-Political Party Bar Plot

It can be observed that about 40% of the tweets examined deal with topics related to the political party called "SYRIZA". The New Democracy (ND) party comes in second, taking about 32% of the conversation on Twitter. The collection also contains references to "ELL_LYSI", which accounts for a little over 2% of the total and "PASOK," which accounts for a noteworthy 14%. Remarkably, the collection also includes references to "DIEM," but with a lesser frequency—less than 5%. At roughly 8%, the Communist Party of Greece (KKE) holds a significant portion.

To further our investigation, we create two separate bar plots, each of which highlights a different aspect of our dataset. The first bar plot (3) provides an in-depth comprehension of the distribution of neutral, positive, and negative attitudes inside each individual political party. At the same time, the second bar plot (4) turns our attention to a wider view, showing how tweets are distributed throughout the dataset in various sentiment categories.

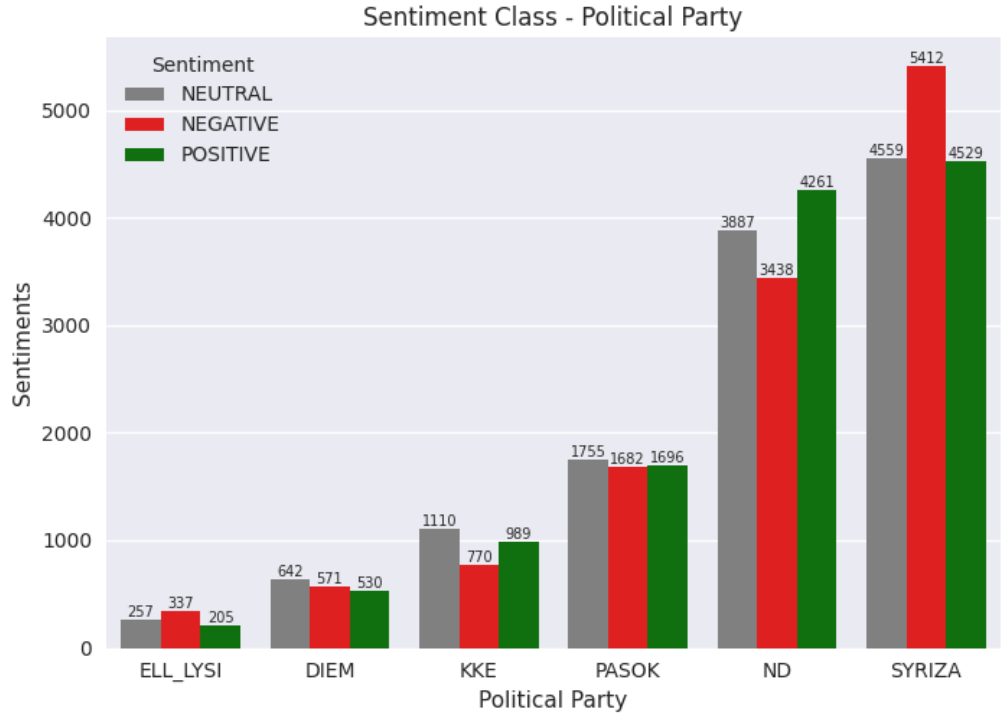

Figure 3: Sentiments-Political Party Bar Plot

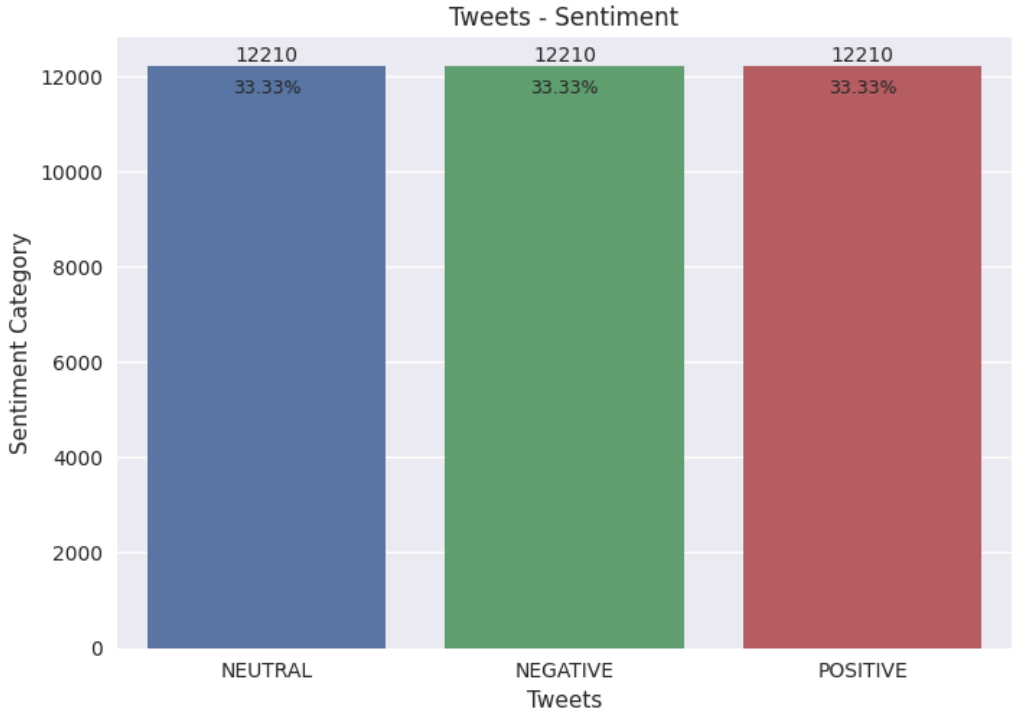

Figure 4: Tweets-Sentiment Bar Plot

Analysis reveals that while sentiment categories within each political party are not

exactly balanced, they are admirably close to being balanced. This adds to an overall balanced dataset along with a fairly distributed distribution of tweets within every sentiment category. Fairness and ease of model training are two benefits of a balanced dataset. Unbiased exposure to every class ensures that no class has an undue influence on models during training. Finally, evaluation criteria that facilitate easy interpretation, such as accuracy, become more trustworthy measures of model performance.

2.3. Data partitioning for train, test and validation

The data sets chosen were the ones that the instructors provided. This decision was made in order to prevent any further complications regarding the development of the code. A ratio of 0.156 is attained by utilizing data from the validation set (5232) and training set (33630).

2.4. Vectorization/Tokenization

In our previous assignment, we looked at a variety of text representation methods to convert textual data into numerical formats appropriate for machine learning applications, including the TF-IDF vectorizer, CountVectorizer, HashVectorizer and Word embeddings. Every approach has advantages and disadvantages of its own, providing different insights on how best to convey the main ideas of the supporting material.

However, for this project, we opted for a different approach and employed tokenization due to the specific requirements of BERT models. Unlike traditional neural networks, BERT models necessitate data to be presented in a tokenized format. Tokenization involves breaking down text into smaller units, such as words or subwords, and assigning each unit a unique identifier or token. This process allows BERT models to effectively understand the contextual relationships within the text. Therefore, by utilizing tokenization, we ensured that our data was appropriately structured and compatible with the input format required by BERT models, enabling them to perform optimally for our project's objectives.

Furthermore, I also used padding to ensure that all sentences within each batch have the same size. This uniformity is crucial because language model, including BERT, typically operate on fixed-size input sequences. By padding shorter sentences with special tokens to match the length of the longest sentence in the batch, we maintain consistency in input dimensions across the entire dataset. This uniformity optimizes computational efficiency during training and inference, as it allows for streamlined processing of batches without the need for dynamic adjustments to accommodate varying sentence lengths. Consequently, padding facilitates smoother and more efficient training of the model, ultimately enhancing its performance and effectiveness in handling diverse text inputs.

3. Generic BERT Language Model

I developed a customizable class called GenericBERT that generates the optimal BERT model based on specified hyperparameters—hidden size, dropout ratio, and choice between GreekBERT or DistilGREEK-BERT. This class ensures simplicity and efficiency by returning a compact model tailored to the user's needs. The design promotes ease

of implementation and flexibility, enabling seamless integration into diverse applications.

4. Algorithms and Experiments

The classification task in the conducted experiments was carried out with an emphasis on grouping data into three separate classes. A variety of reccurent neural networks with distinct architectures and configurations were utilized. The main goal was to assess how neural network architecture affected the classification accuracy for each of the specified classes.

4.1. Hyper-parameter tuning

During the course of my experiments, I elected to employ the Optuna framework, a tool introduced by our instructors, enabling seamless execution and evaluation of trials accompanied by valuable insights pertaining to optimization processes and overall model behavior. Among the varied parameters explored throughout this investigation were:

- Hidden Size
- Dropout Probabilities
- Epochs
- Learning Rate
- Batch Size
- Maximum Length (for padding)

Utilization of this comprehensive suite of variables allowed for thorough examination of individual impacts on model efficiency and effectiveness, thereby fostering enhanced understanding of optimal configuration strategies tailored specifically towards addressing complex sequence prediction challenges.

4.2. Evaluation

A thorough set of six measures, each providing a unique perspective on the model's performance, will be included in the evaluation process. These measures are f1-score, accuracy, recall, precision, average loss and time taken. the F1-score is given particular consideration since it may effectively balance recall and precision. The F1-score is calculated as follows:

$$F1\ Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{1}$$

In addition, a Confusion Matrix and a Receiver Operating Characteristic (ROC) curve were used for more in-depth study of the top model found using these measures.

4.3. Optuna Study For GreekBERT

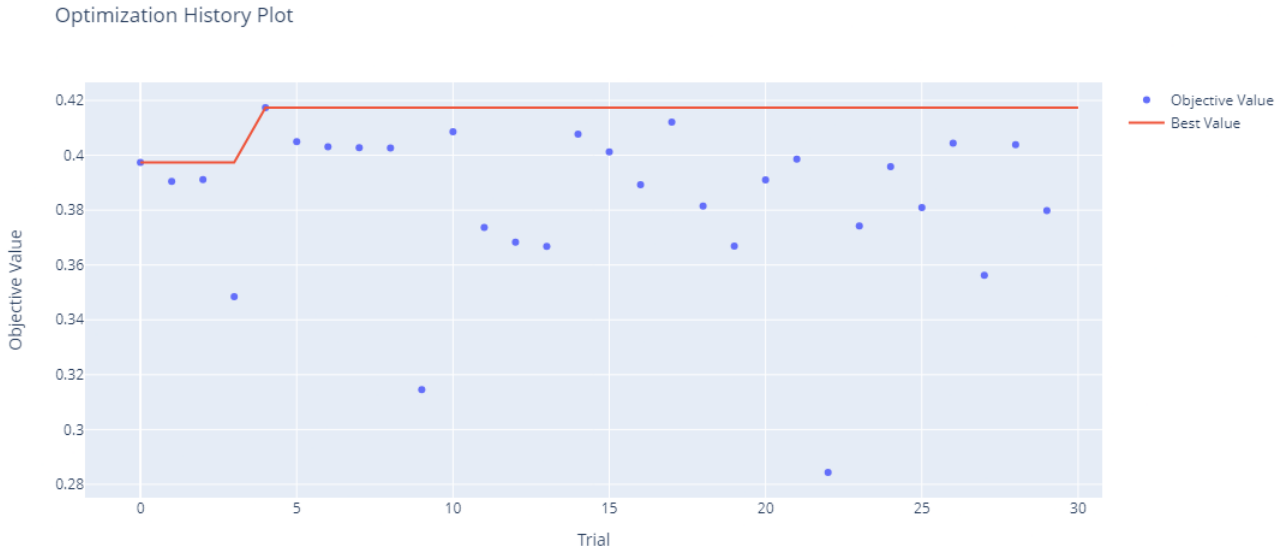

Figure 5: Optuna Optimization History Plot

Examining the Optuna's optimization history plot reveals a notable trend, with the majority of trials yielding objective values between 0.36 to 0.42. However, it's worth noting that a solitary trial achieving a value of 0.42 was swiftly rejected due to excessive overfitting. Despite this outlier, the concentration of trials within this range suggests a generally positive outcome, particularly when contrasted with previous assignments involving RNNs. In those instances, the majority of objective values fell below 0.2, indicating prevalent issues of over/under-fitting and overall poor performance. Thus, the prevalence of trials within the 0.36 to 0.42 range signifies a marked improvement and suggests that the majority of models generated through Optuna's optimization process exhibit promising effectiveness for the task at hand. Overall, the optimization history plot serves as a powerful diagnostic tool, elucidating the efficacy of employed strategies and shedding light on the impact of varying hyperparameters on model fitness. By providing a holistic perspective on the evolution of candidate evaluations, researchers gain essential perspectives concerning the merit of different configurations, informing decisions surrounding optimal selections and driving continuous improvement initiatives.

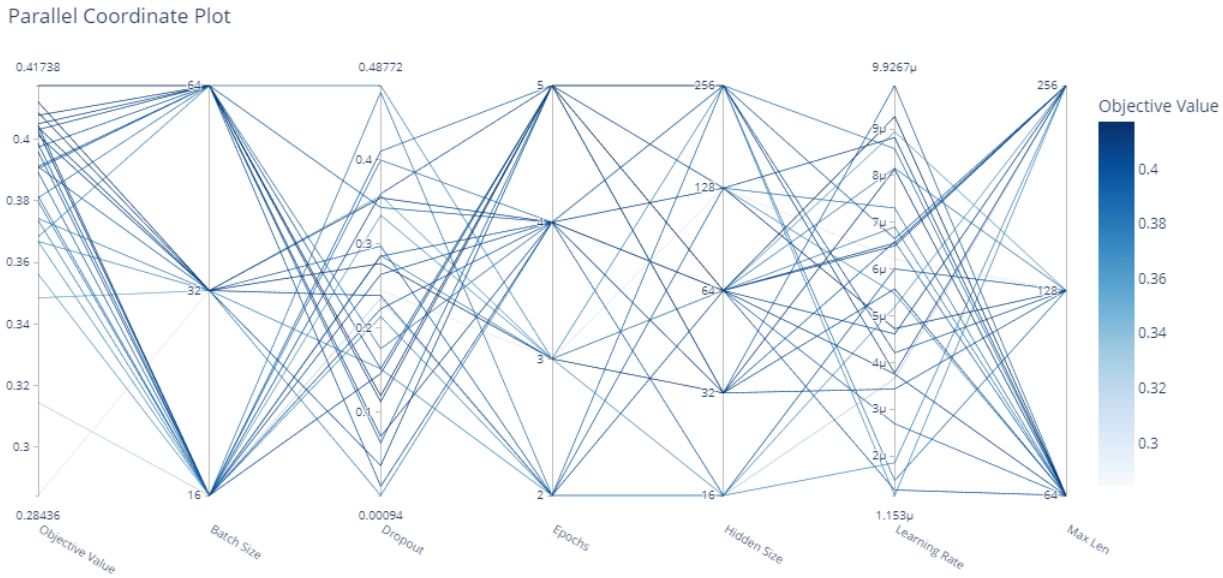

Figure 6: Optuna Parallel Coordinate Plot

The parallel coordinate plot provides a visual representation of the model's performance, with darker lines indicating superior results. Upon scrutiny, several notable observations emerge. Among the evaluated batch sizes, 16 emerges as the most optimal, followed by 64 and then 32, indicating a clear hierarchy in performance. Interestingly, all dropout ratio values yield similar outcomes, suggesting a lack of significant impact on model effectiveness. In terms of epochs, 5 stands out as the best-performing value, striking a balance between avoiding overfitting and underfitting. Regarding hidden size, both 256 and 64 exhibit comparable performance, suggesting their suitability as ideal values. Notably, darker lines cluster around learning rate values between 6μ and 7μ, indicating their superiority. Finally, for maximum length padding, the size of 64 outperforms 256, demonstrating its slightly better efficacy in the model's performance. Overall, the parallel coordinate plot proves instrumental in deciphering interdependencies among hyperparameters and quantifying their net contributions to the model's predictive accuracy, thus empowering informed decisionmaking during iterative refinement stages.

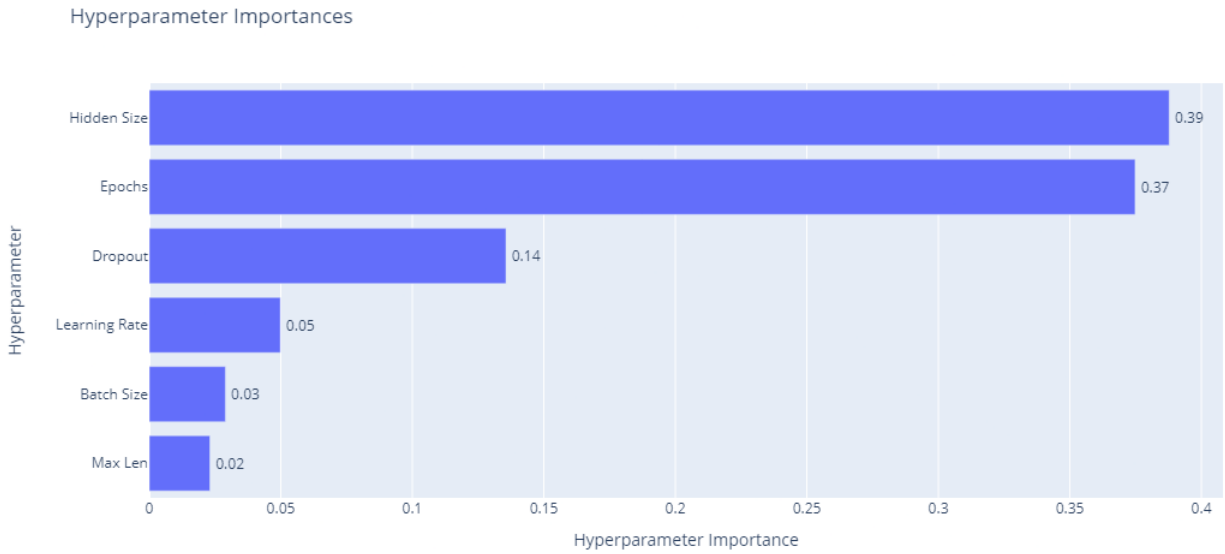

Figure 7: Optuna Hyperparameter Importances

The hyperparameter importance plot reveals critical insights into the factors driving the model's performance. With a substantial importance value of 0.39, hidden size emerges as the most influential parameter, indicating its pivotal role in shaping the model's effectiveness. Following closely behind, epochs rank second with a significance value of 0.37, highlighting the crucial balance required in training iterations to optimize results. Dropout, although important with a value of 0.14, falls short of the impact exerted by hidden size and epochs. Learning rate follows with a comparatively lower importance value of 0.05, suggesting its relevance but lesser influence on model performance. Batch size and max len exhibit even lower importance values of 0.03 and 0.02, respectively, indicating their relatively minor impact on overall effectiveness. In summary, this analysis underscores the paramount importance of hidden size and epochs in driving model performance, with other hyperparameters playing increasingly less influential roles in comparison. Each of these components contributes uniquely towards regulating information flow, managing complexity, and enriching representations. Their logical configuration supports the model's ability to learn meaningful abstractions, resist overfitting tendencies, and maintain stability during training. Collectively, this comprehensive assessment of hyperparameter importances offers valuable insights regarding the primary levers influencing model performance, guiding future efforts aimed at refining architectural choices and calibrating operational settings.

Optuna Best GreekBERT Model

Hidden Size:	256
Dropout:	0.11216215886569281
Epochs:	5
Learning Rate:	3.7712893289199845e-06
Batch Size:	64
Max Length:	256

Table 1: Architecture

Time Taken:	1949.149
Average Loss:	1.0659800120970098
F1-Score:	0.4077274799346924
Accuracy:	0.4082568883895874
Recall:	0.4082568883895874
Precision:	0.4085540175437927

Table 2: Metrics

Training classification report	precision	recall	f1-score	support
NEGATIVE	0.45	0.43	0.44	12210
NEUTRAL	0.44	0.42	0.43	12210
POSITIVE	0.44	0.50	0.47	12210
accuracy			0.45	36630
macro avg	0.45	0.45	0.45	36630
weighted avg	0.45	0.45	0.45	36630

Table 3: Train Classification Report

Validation classification report	precision	recall	f1-score	support
NEGATIVE	0.41	0.38	0.40	1744
NEUTRAL	0.41	0.39	0.40	1744
POSITIVE	0.40	0.45	0.43	1744
accuracy			0.41	5232
macro avg	0.41	0.41	0.41	5232
weighted avg	0.41	0.41	0.41	5232

Table 4: Val Classification Report

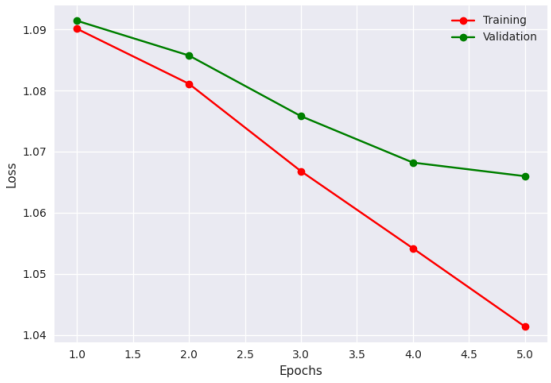

Figure 8: Learnig Curve (Loss)

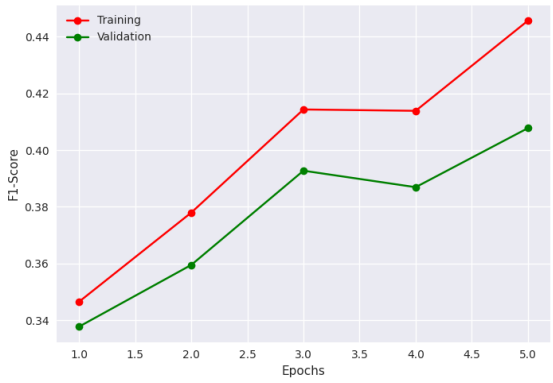

Figure 9: Learnig Curve (F1-Score)

Following experimentation, the Optuna's best model showcased promising performance metrics. Analysis of the loss curve indicates a steady decline in both training and validation losses, typically considered favorable indicators of model learning. While some may raise concerns regarding potential overfitting due to the gap between the two curves, I argue that the discrepancy falls within acceptable margins. This assertion is substantiated by the parallel increase observed in the F1 curves, which suggest a balanced improvement in performance across both sets. Moreover, examination of classification reports for both training and validation datasets corroborates the model's effectiveness. Achieving results significantly exceeding previous assignments, where performance struggled to surpass 40%, underscores the success of this model. However, it's essential to note the drawback of extended training times, which presents a notable challenge in practical application. Despite this limitation, the model's overall performance remains commendable.

4.4. Optuna Study For DistilGREEK-BERT

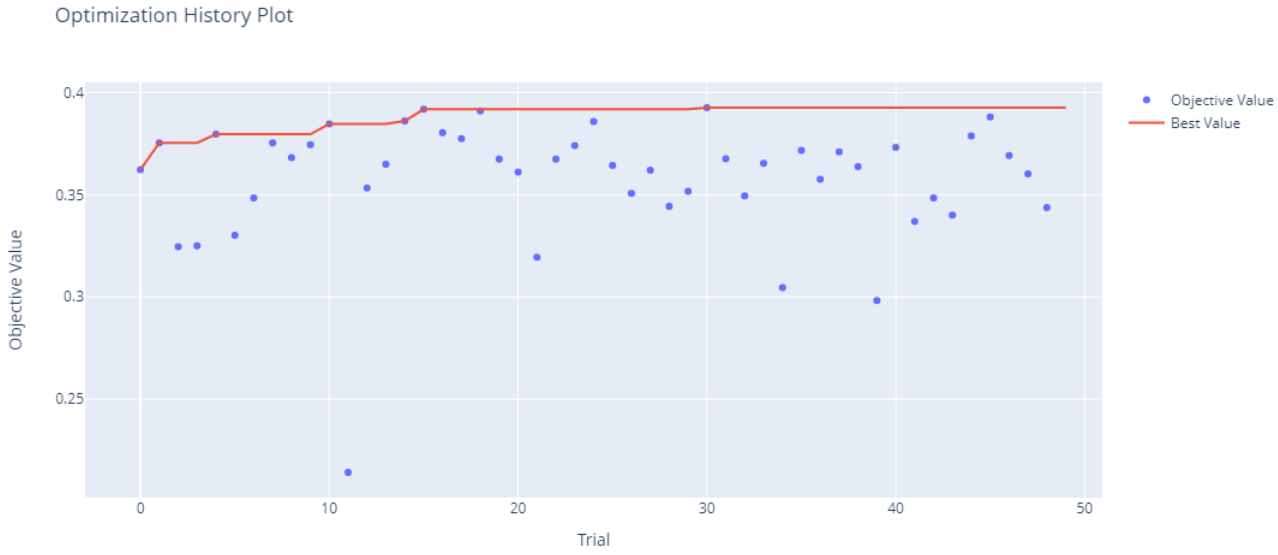

Figure 10: Optuna Optimization History Plot

Upon analyzing the Optuna's optimization history plot, it became apparent that the majority of trials fell within the range of 0.35 to 0.39. Notably, the study did not reach the 40% mark, which holds particular interest. In comparison to GreekBERT, the observed objective values were generally smaller. Despite this, it's noteworthy that even DistilGREEK-BERT achieved superior performance compared to the RNNs study. This observation underscores the significance of transformer-based models like DistilGREEK-BERT in achieving enhanced results over traditional RNN architectures. While the study did not breach the 40% threshold, the performance improvement offered by transformer models signifies promising advancements in natural language processing tasks.

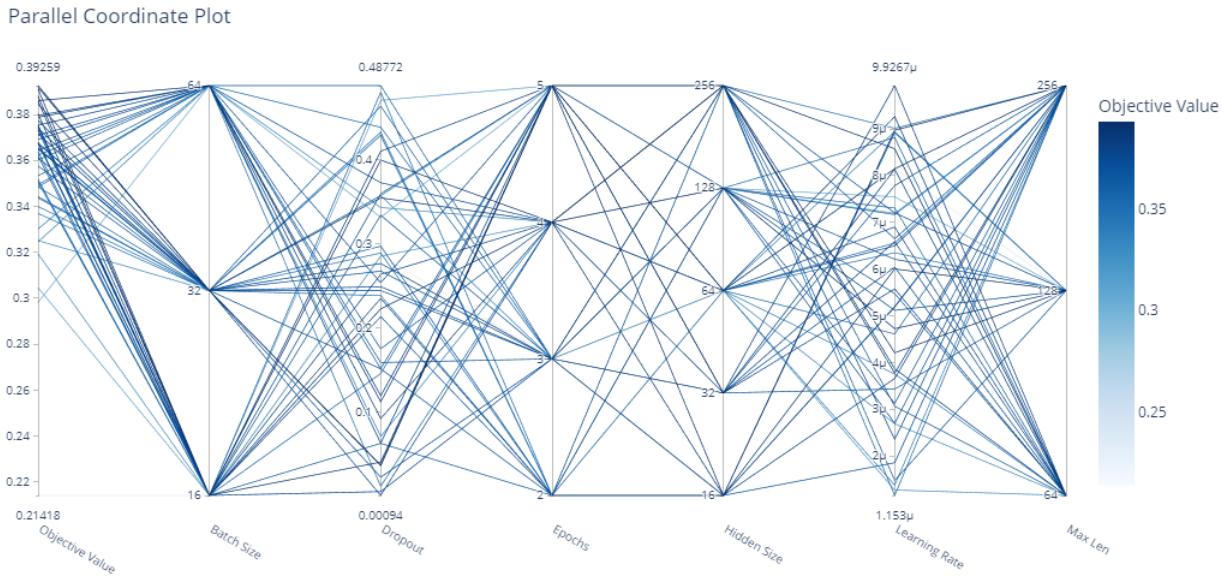

Figure 11: Optuna Parallel Coordinate Plot

Analyzing the parallel coordinate plot provides valuable insights into the performance of various model configurations. The darkness of the lines correlates with the quality of values, indicating that darker lines represent better-performing configurations. In terms of batch size, the plot reveals evenly distributed dark lines, suggesting that all batch size values perform equally well. Dropout, on the other hand, shows negligible dark lines, indicating minimal impact on the trials, like in the previous Optuna study for GreekBERT. Moving to epochs, the value of 5 emerges as optimal, maintaining a balance between overfitting and underfitting. Interestingly, for hidden size, no single value stands out, indicating equality among them. While a slightly darker line appears around the learning rate value of 9µ, there are no other notable observations. Finally, the trend persists in max length for padding, with both 64 and 256 sizes displaying similar effectiveness. Overall, the parallel coordinate plot proves instrumental in deciphering interdependencies among hyperparameters and quantifying their net contributions to the model's predictive accuracy, thus empowering informed decision-making during iterative refinement stages.

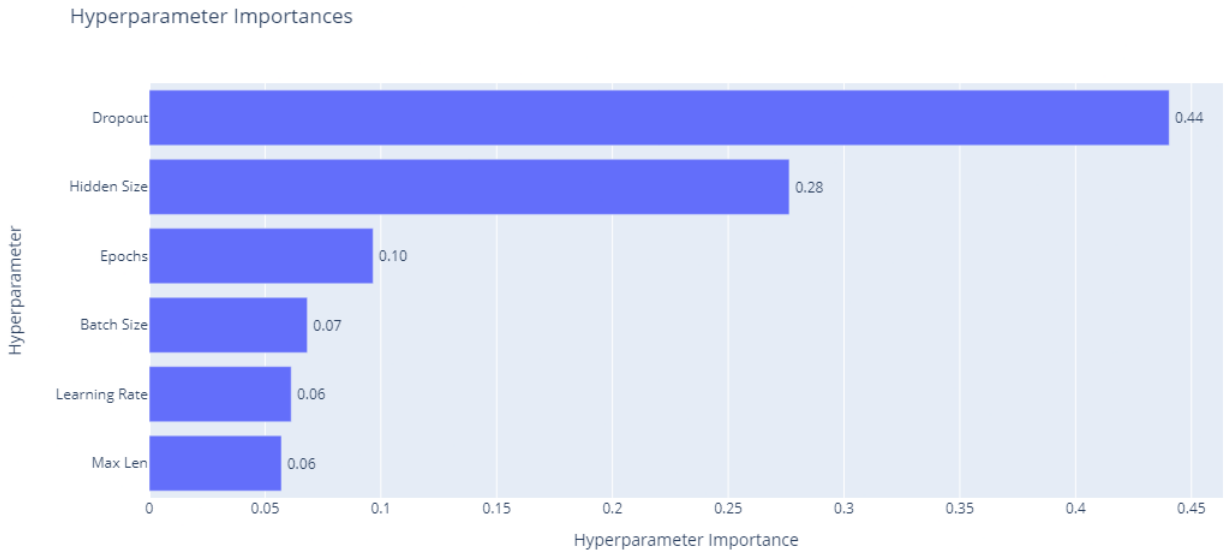

Figure 12: Optuna Hyperparameter Importances

The hyperparameter importance plot reveals crucial insights into the factors driving the model's performance. With a significant importance value of 0.44, dropout emerges as the most influential parameter, indicating its pivotal role in shaping the model's effectiveness. Following behind, hidden size holds substantial importance with a value of 0.28, suggesting its significant impact on model performance. Epochs, though important, exhibit a notably lower importance value of 0.10, indicating their comparatively lesser influence. Batch size, learning rate, and max len follow with importance values of 0.07, 0.06, and 0.06, respectively, suggesting their minor impact on overall effectiveness compared to dropout and hidden size. This analysis underscores the paramount importance of dropout and hidden size in driving model performance, while also recognizing the contributions of other hyperparameters, albeit to a lesser extent. Each of these components contributes uniquely towards regulating information flow, managing complexity, and enriching representations. Their logical configuration supports the model's ability to learn meaningful abstractions, resist overfitting tendencies, and maintain stability during training. Collectively, this comprehensive assessment of hyperparameter importances offers valuable insights regarding the primary levers influencing model performance, guiding future efforts aimed at refining architectural choices and calibrating operational settings.

Optuna Best DistilGREEK-BERT Model

Hidden Size:	16
Dropout:	0.03703912923817959
Epochs:	5
Learning Rate:	8.971553826587668e-06
Batch Size:	32
Max Length:	256

Table 5: Architecture

Time Taken:	1065.490
Average Loss:	1.0830115959411715
F1-Score:	0.3925908803939819
Accuracy:	0.3931574821472168
Recall:	0.3931574821472168
Precision:	0.3938202261924743

Table 6: Metrics

Training classification report	precision	recall	f1-score	support
NEGATIVE	0.41	0.36	0.38	12210
NEUTRAL	0.39	0.43	0.41	12210
POSITIVE	0.41	0.41	0.41	12210
accuracy			0.40	36630
macro avg	0.40	0.40	0.40	36630
weighted avg	0.40	0.40	0.40	36630

Table 7: Train Classification Report

Validation classification report	precision	recall	f1-score	support
NEGATIVE	0.40	0.36	0.38	1744
NEUTRAL	0.38	0.43	0.41	1744
POSITIVE	0.40	0.39	0.39	1744
accuracy			0.39	5232
macro avg	0.39	0.39	0.39	5232
weighted avg	0.39	0.39	0.39	5232

Table 8: Val Classification Report

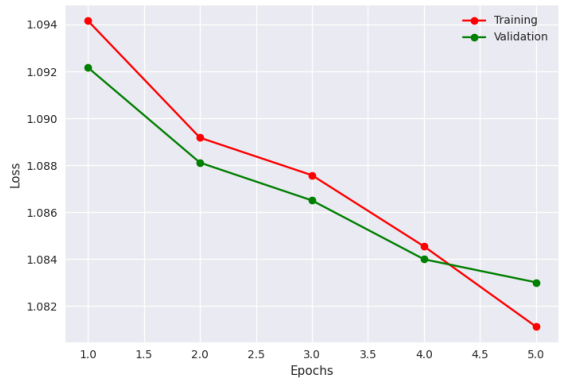

Figure 13: Learnig Curve (Loss)

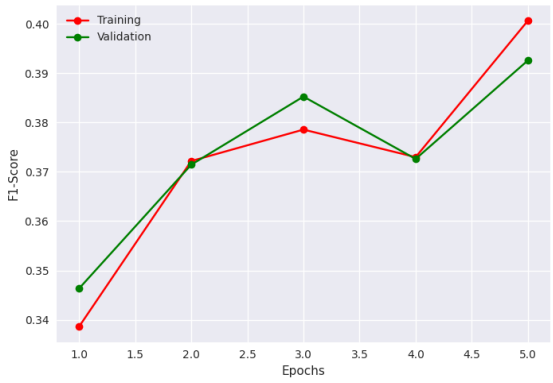

Figure 14: Learnig Curve (F1-Score)

The Optuna's top-performing model demonstrates impressive results on both the training and validation sets, as evidenced by the near-identical decline in their respective loss curves. This remarkable similarity signals that the model has achieved a delicate equilibrium between capturing underlying patterns in the training data and retaining the capacity to perform accurately when faced with previously unencountered samples. Additionally, the f1 curves display a harmonious increase, reinforcing the notion that the model excels in identifying positive cases (true positives) versus negative ones (false negatives), regardless of whether they originate from the training or validation dataset.

5. Best Models

5.1. Final Model – Logistic Regression

The final Logistic Regression model consists of the following:

- TF-IDF Vectorizer
- C = 0.001
- solver = liblinear
- penalty = l2

Performance

- Time Taken: 3.70
- F1-Score: 0.3778712561197638
- Accuracy: 0.3843646875987297
- Recall: 0.3844635466411971
- Precision: 0.3887267312255672

Below are the represenations of the ROC curve, learning curve and confusion matrix for the final mode:

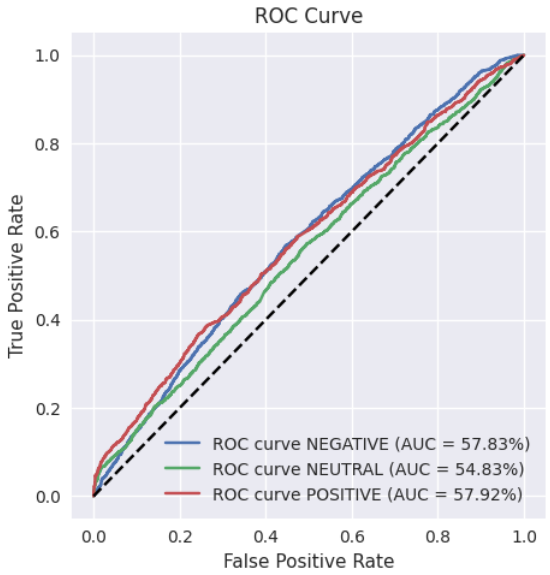

Figure 15: Final Model LR Roc Curve

In Class 1 (NEGATIVE), the AUC is more than 50%, which indicates that the ROC curve is comparatively superior than random chance. That isn't very high, though, suggesting that the model can only slightly outperform chance in distinguishing between instances of the NEGATIVE class. The AUC for Class 2 (NEUTRAL) is 54.83%, which is somewhat greater than 50%. This implies that although there is not much discrimination, the model performs marginally better for the NEUTRAL class than random chance. Class 3 (POSITIVE): The model can reasonably discriminate between cases for the POSITIVE class, according to the AUC of 57.92%. It's not a very strong performance, but it's better than chance.

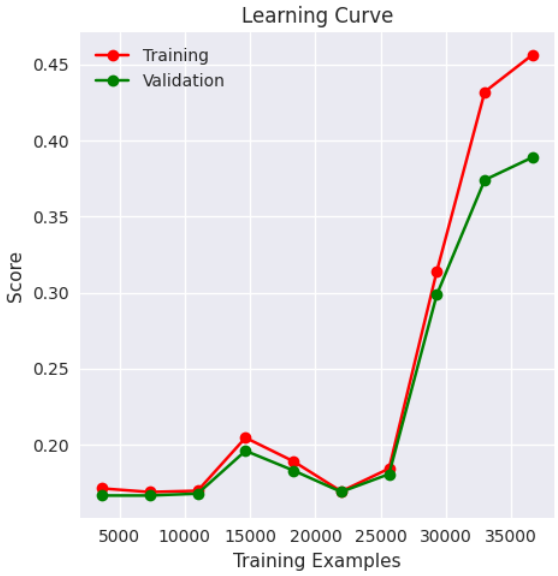

Figure 16: Final Model LR Learning Curve

The learning curve started off slightly, which was evidence of the early lack of skill in the training and validation datasets. But as the iterations went on, there was a noticeable increase. The training curve showed a consistent upward trend, indicating that the model was able to identify more intricate patterns in the data. Simultaneously, the validation curve had an upward trend, indicating the model's excellent applicability to previously unreported data.

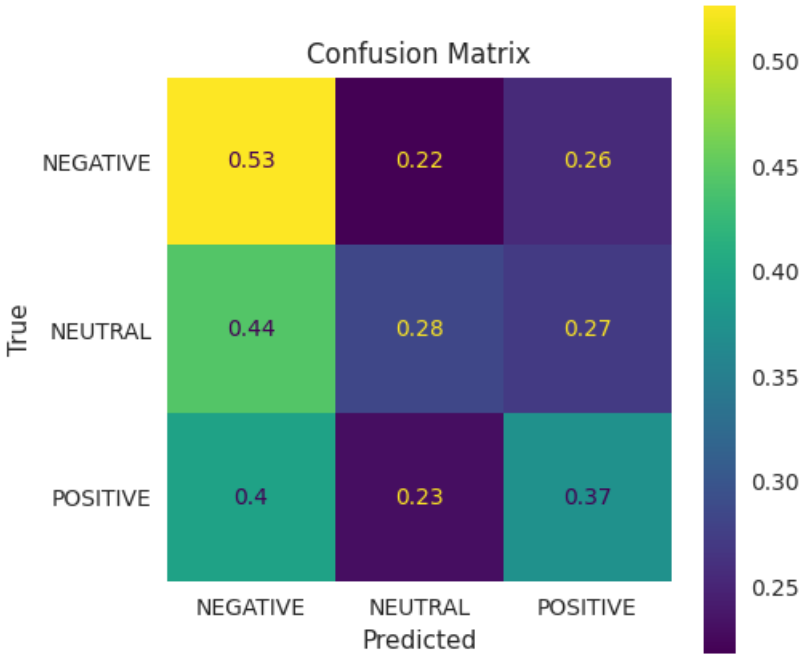

Figure 17: Final Model LR Confusion Matrix

A confusion matrix consists of the following:

- The true NEGATIVE is represented by the first row.
- The true NEUTRAL is represented by the second row.

- The true POSITIVE is represented by the third row.
- The predicted NEGATIVE is represented by the first column.
- The predicted NEUTRAL is represented by the second column.
- The predicted POSITIVE is represented by the third column.

The diagonal values are representing the correct prediction for each class:

- 53% for NEGATIVE
- 28% for NEUTRAL
- 37% for POSITIVE

5.2. Final Model – Feed Forward Neural Network

The final Feed Forward Neural Network model consists of the following:

- Word Embedding
- 3 Hidden Layers
  - 1st hidden layer has size 128
  - 2nd hidden layer has size 128
  - 3rd hidden layer has size 16
- Activation Function = ReLU
- Epochs = 10
- Optimizer = SGD
- Learning Rate = 0.01

Performance

- Time Taken: 40.929
- Average Loss: 1.075634155798396
- F1-Score: 0.392484903335571
- Accuracy: 0.401949554681777
- Recall: 0.401949554681777
- Precision: 0.407101511955261

Below are the represenations of the ROC curve, learning curve and confusion matrix for the final mode:

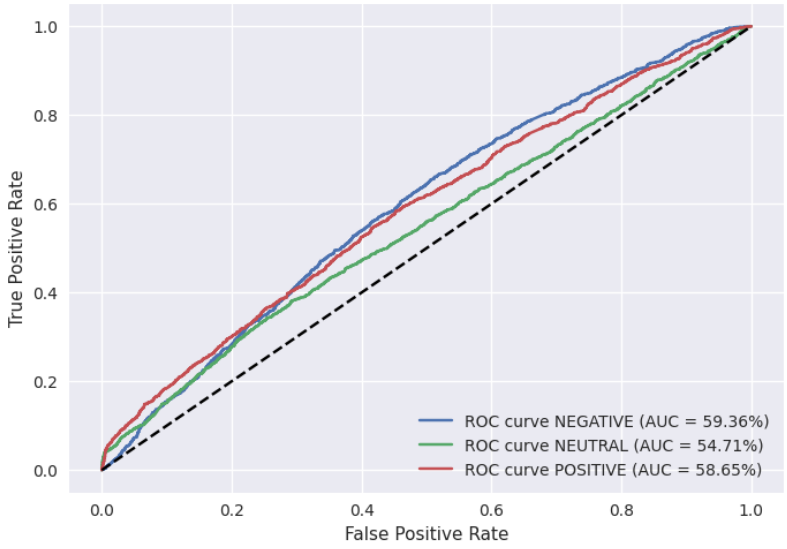

Figure 18: Final Model Roc Curve

The negative class has an AUC (Area Under the Curve) of 59.36%. This means that the classifier performs only slightly better than random guessing when distinguishing between negative samples and the other two classes. The ROC curve for the negative class is relatively flat, indicating that the TPR does not increase much as the FPR increases. The neutral class has an AUC of 54.71%. This is worse than random guessing, indicating that the classifier struggles to distinguish between neutral samples and the other two classes. The ROC curve for the neutral class is also relatively flat, but it is shifted downwards compared to the negative class curve. The positive class has an AUC of 58.65%. This is slightly better than random guessing, but still not a very strong performance. The ROC curve for the positive class is slightly curved upwards, indicating that the TPR increases more rapidly than the FPR as the threshold is decreased. Overall, the ROC curves for all three classes suggest that the classifier is not performing very well. The AUC values are all relatively low, and the ROC curves do not show much separation between the TPR and FPR. This suggests that there may be room for improvement in the classifier's design or training process.

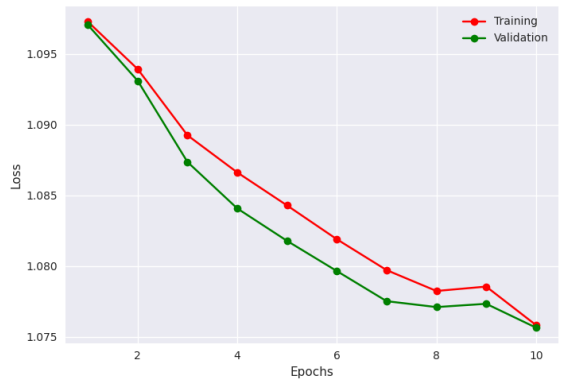

Figure 19: Learnig Curve (Loss)

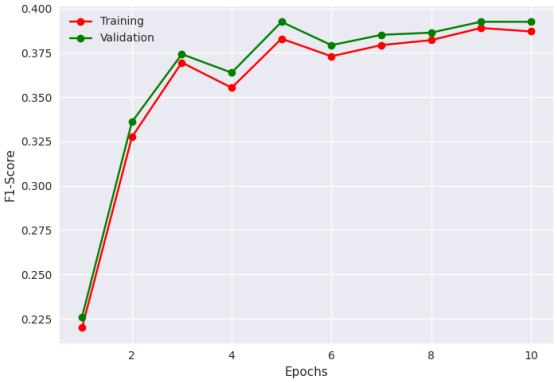

Figure 20: Learnig Curve (F1-Score)

Based on the learning curves, it appears that the model is improving as the number of epochs increases, with both the training and validation loss decreasing and the F1-score increasing for both train and validation sets. The learning curve for the loss shows that as the number of epochs increases, both the training and validation loss decrease, indicating that the model is learning to fit the data better over time. The fact that the validation loss is also decreasing suggests that the model is not overfitting to the training data, which is a good sign. Similarly, the learning curve for the F1-score shows that it is increasing for both the training and validation sets as the number of epochs increases. An increasing F1-score indicates that the model is becoming more accurate in its predictions, and the fact that this is true for both the training and validation sets suggests that the model is not overfitting. Overall, the learning curves suggest that the model is improving over time and is not overfitting to the training data. This is a positive sign and indicates that the model is well-designed and properly tuned.

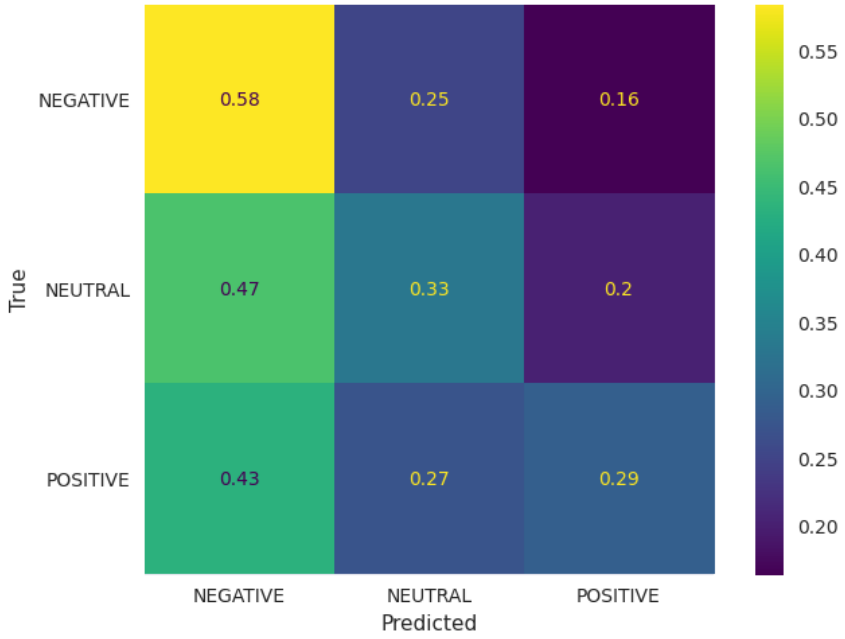

Figure 21: Final Model Confusion Matrix

A confusion matrix consists of the following:

- The true NEGATIVE is represented by the first row.
- The true NEUTRAL is represented by the second row.
- The true POSITIVE is represented by the third row.
- The predicted NEGATIVE is represented by the first column.
- The predicted NEUTRAL is represented by the second column.
- The predicted POSITIVE is represented by the third column.

The diagonal values are representing the correct prediction for each class:

- 58% for NEGATIVE
- 33% for NEUTRAL
- 29% for POSITIVE

5.3. Final Model – Recurrent Neural Network

The final Recurrent Neural Network model consists of the following:

- Vectorizer: Word Embedding
- Cell: LSTM
- Hidden Size: 64

- Stacked Layers: 3
- Dropout: 0
- Skip: True
- Attention: True
- Clip: 0
- Epochs: 6
- Optimizer: Adam
- Learning Rate: 0.002752755011361541

Performance

- Time Taken: 116.504
- Average Loss: 1.07745507897222
- F1-Score: 0.39271280169487
- Accuracy: 0.402140676975250
- Recall: 0.402140676975250
- Precision: 0.404058635234832

Below are the represenations of the ROC curve, learning curve and confusion matrix for the final mode:

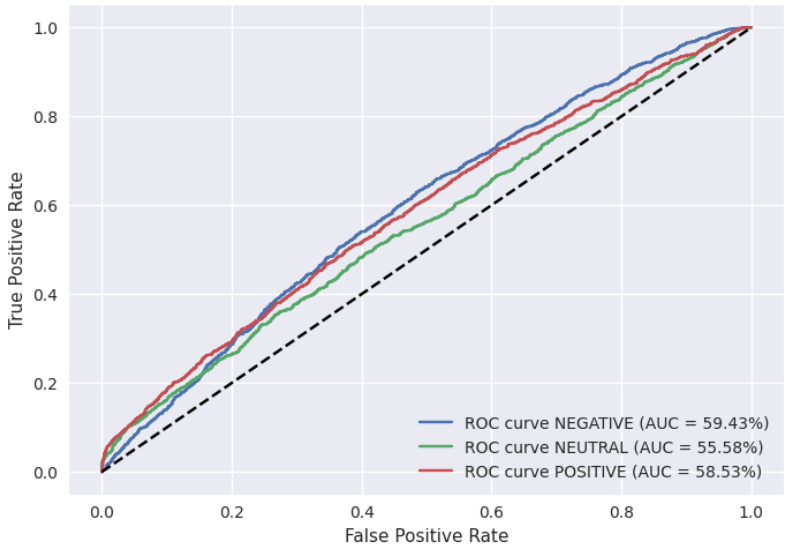

Figure 22: Final Model Roc Curve

A ROC (Receiver Operator Characteristics) curve illustrates the relationship between true positive rates (TPR) and false positive rates (FPR) across various classification thresholds. With the given AUC values for negative (59.43%), neutral (55.58%), and positive (58.53%) classes, here's a brief interpretation of the resulting ROC curves:

- Negative class: The AUC of 59.43% indicates that the classifier performs moderately well when distinguishing between negative samples and others. The ROC curve for the negative class would exhibit a gradual inclination, reflecting the increasing TPR as the FPR grows. However, due to the relatively low AUC, the curve wouldn't rise too sharply, demonstrating limited ability to separate negative samples from others.
- Neutral class: The AUC of 55.58% suggests that the classifier struggles to identify neutral samples. Consequently, the ROC curve for the neutral class would be relatively flat, with little change in TPR as the FPR increases. This indicates that the classifier cannot effectively distinguish between neutral samples and other classes.
- Positive class: The AUC of 58.53% implies that the classifier performs marginally better than the neutral class when identifying positive samples. The ROC curve for this class would exhibit a mild upward trend, indicating that the TPR increases more rapidly than the FPR as the threshold is adjusted. Nevertheless, the curve won't rise dramatically since the AUC remains below 60%.

In summary, the ROC curves for all three classes reveal that the classifier isn't performing optimally which is expected due to the miss-labeled dataset. The low AUC values and flat ROC curves for the neutral class suggest that improvements should focus on enhancing the classifier's capacity to recognize neutral samples. Additionally, further tuning the classifier's parameters and exploring alternative models might help boost performance for all classes.

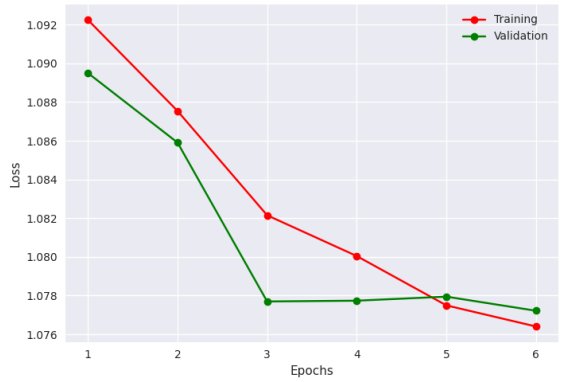

Figure 23: Learnig Curve (Loss)

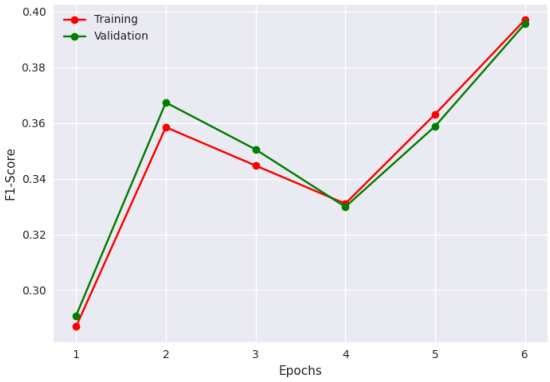

Figure 24: Learnig Curve (F1-Score)

Based on the learning curve analysis, the model exhibits continuous improvement as the number of epochs progresses. Both training and validation losses diminish while the F1-score increases for both training and validation sets. The learning curve for loss demonstrates that as the number of epochs advances, both training and validation losses decline, suggesting enhanced fitting to the data. The consistent reduction in validation loss confirms that the model is not suffering from overfitting. Similarly, the learning curve for the F1-score highlights growth for both training and validation sets as the number of epochs increases, implying increased accuracy in prediction. The concurrent enhancement for both training and validation sets denotes that the model is not

prone to overfitting. Overall, the learning curves convey a positive message regarding the model's development and its resistance against overfitting, thus confirming that the model is well-designed and appropriately fine-tuned.

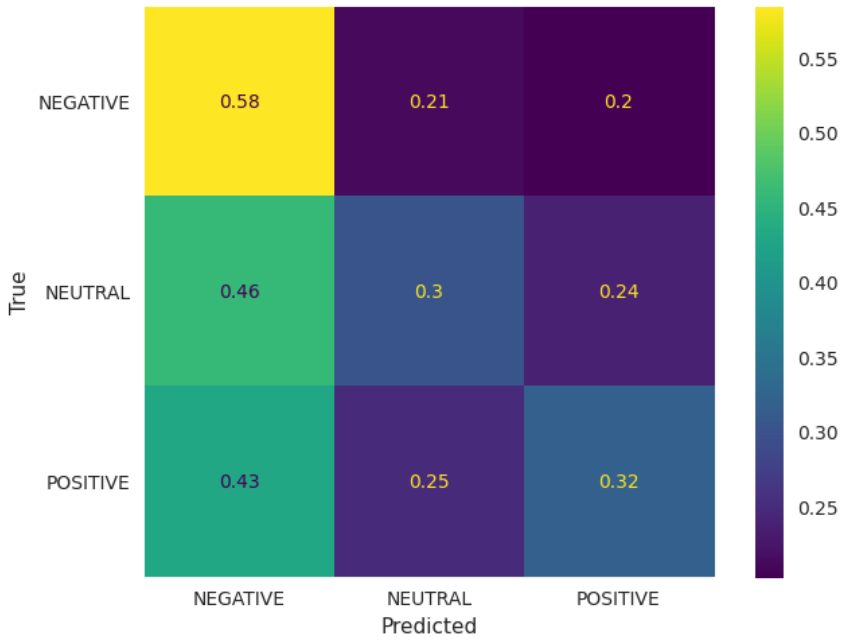

Figure 25: Final Model Confusion Matrix

A confusion matrix consists of the following:

- The true NEGATIVE is represented by the first row.
- The true NEUTRAL is represented by the second row.
- The true POSITIVE is represented by the third row.
- The predicted NEGATIVE is represented by the first column.
- The predicted NEUTRAL is represented by the second column.
- The predicted POSITIVE is represented by the third column.

The diagonal values are representing the correct prediction for each class:

- 58% for NEGATIVE
- 30% for NEUTRAL
- 32% for POSITIVE

5.4. Final Model – GreekBERT

The final GreekBERT model consists of the following:

- Hidden Size: 256
- Dropout: 0.11216215886569281
- Epochs: 5
- Optimizer: AdamW
- Learning Rate: 3.7712893289199845e-06
- Batch Size: 64
- Max Length: 256

Performance

- Time Taken: 1948.812
- Average Loss: 1.0659800120970098
- F1-Score: 0.4077274799346924
- Accuracy: 0.4082568883895874
- Recall: 0.4082568883895874
- Precision: 0.4085540175437927

Below are the represenations of the ROC curve, learning curve and confusion matrix for the final mode:

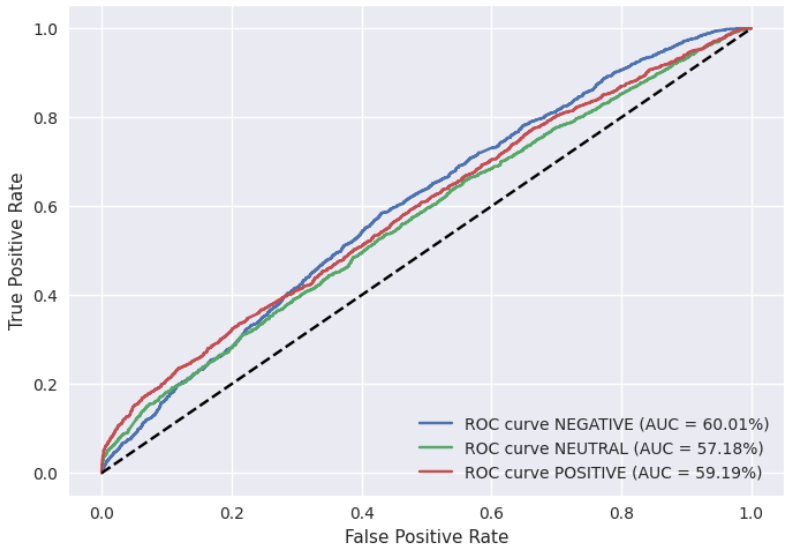

Figure 26: Final Model Roc Curve

A ROC (Receiver Operator Characteristics) curve illustrates the relationship between true positive rates (TPR) and false positive rates (FPR) across various classification thresholds. With the given AUC values for negative (60.01%), neutral (57.18%), and positive (59.19%) classes, here's a brief interpretation of the resulting ROC curves:

- Negative class: The AUC for the negative class is 60.01%, showing that the classifier does relatively well in differentiating negative data from others. The ROC curve for the negative class would be gradually inclined, representing the growing TPR as the FPR increases. However, due to the relatively low AUC, the curve did not climb too quickly, indicating a limited capacity to distinguish negative samples from others.
- Neutral class: The neutral class has an AUC of 57.18%, indicating that the classifier struggles to recognize neutral data. As a result, the ROC curve for the neutral class will be very flat, with minimal change in TPR as the FPR grows. This suggests that the classifier is unable to successfully discriminate between neutral samples and other classes.
- Positive class: The AUC for the positive class is 59.19%, indicating that the classifier performs somewhat better than the neutral class for detecting positive data. This class's ROC curve would show a little upward trend, suggesting that when the threshold is changed, the TPR grows faster than the FPR. Nonetheless, the curve will not climb considerably because the AUC is still less than 60%.

In summary, the ROC curves for all three classes reveal that the classifier isn't performing optimally which is expected due to the miss-labeled dataset. The low AUC values and flat ROC curves for the neutral class suggest that improvements should focus on enhancing the classifier's capacity to recognize neutral samples. Additionally, further tuning the classifier's parameters and exploring alternative models might help boost performance for all classes.

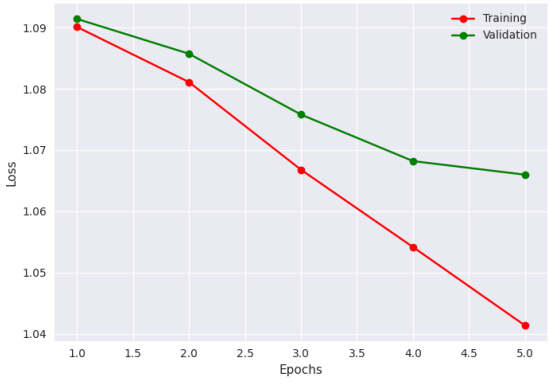

Figure 27: Learnig Curve (Loss)

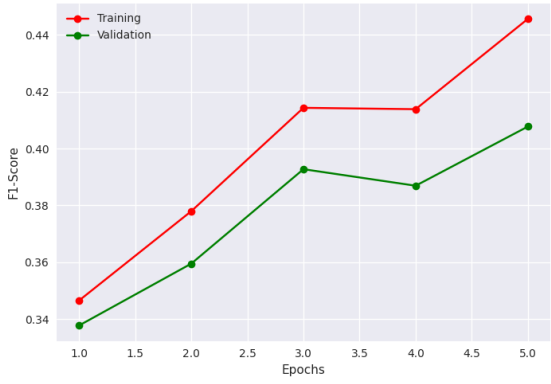

Figure 28: Learnig Curve (F1-Score)

Based on the learning curve analysis, it is evident that the model's performance improves continuously as the number of epochs increases. This can be observed in the declining trends of both training and validation losses, while the F1-score shows an upward trajectory for both training and validation sets. The learning curve for loss

reveals that as the number of epochs advances, both training and validation losses decrease, indicating a better fit to the data. Despite someone potentially arguing that the model is overfitting due to the slower decline in validation loss compared to training loss, the gap between the two losses is within acceptable margins. This is further supported by the classification reports that we analyzed earlier. On the other hand, the learning curve for the F1-score demonstrates growth for both training and validation sets, suggesting an improvement in prediction accuracy. The concurrent enhancement of both training and validation sets indicates that the model is not overfitting. Overall, the learning curves convey a positive message regarding the model's development and its ability to resist overfitting, confirming that the model is well-designed and appropriately fine-tuned.

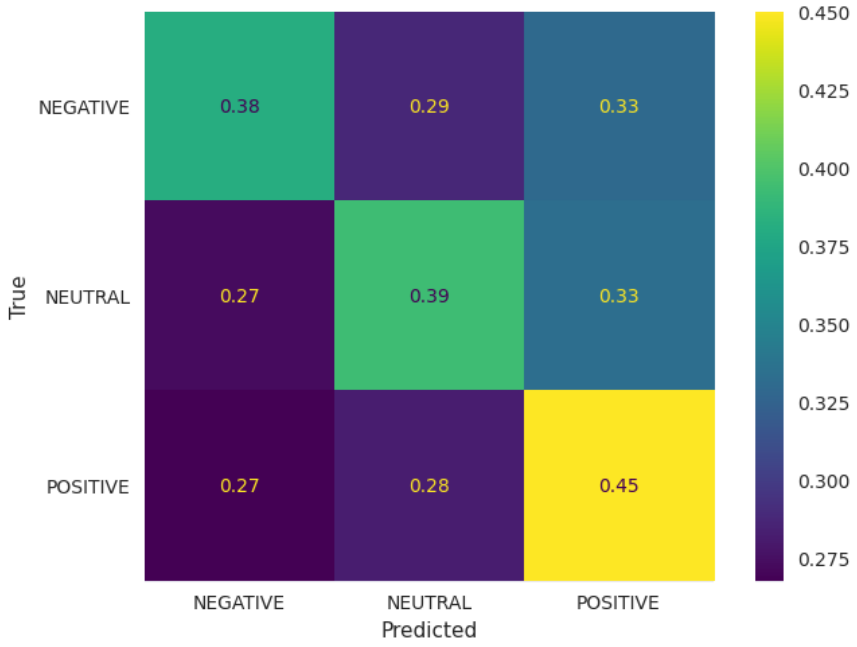

Figure 29: Final Model Confusion Matrix

A confusion matrix consists of the following:

- The true NEGATIVE is represented by the first row.
- The true NEUTRAL is represented by the second row.
- The true POSITIVE is represented by the third row.
- The predicted NEGATIVE is represented by the first column.
- The predicted NEUTRAL is represented by the second column.
- The predicted POSITIVE is represented by the third column.

The diagonal values are representing the correct prediction for each class:

- 38% for NEGATIVE

- 39% for NEUTRAL
- 45% for POSITIVE

5.5. Final Model – DistilGREEK-BERT

The final DistilGREEK-BERT model consists of the following:

- Hidden Size: 16
- Dropout: 0.03703912923817959
- Epochs: 5
- Optimizer: AdamW
- Learning Rate: 8.971553826587668e-06
- Batch Size: 32
- Max Length: 256

Performance

- Time Taken: 1065.542
- Average Loss: 1.0830115959411715
- F1-Score: 0.3925908803939819
- Accuracy: 0.3931574821472168
- Recall: 0.3931574821472168
- Precision: 0.3938202261924743

Below are the represenations of the ROC curve, learning curve and confusion matrix for the final mode:

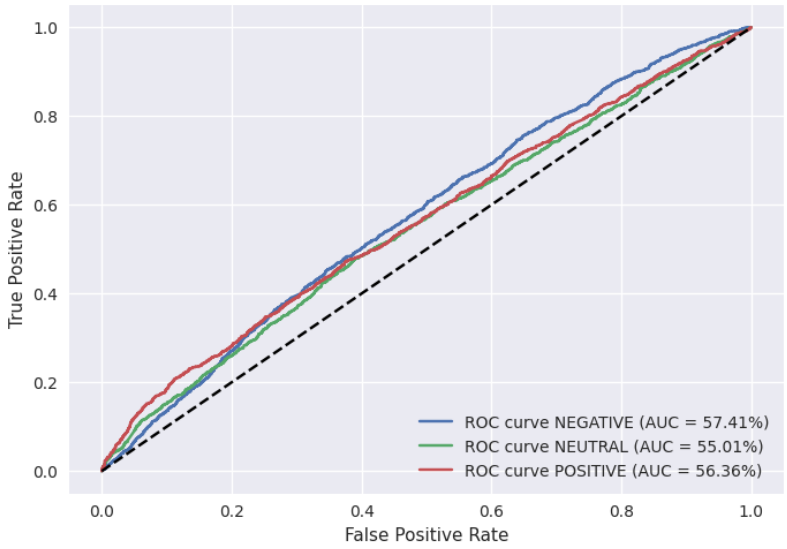

Figure 30: Final Model Roc Curve

A ROC (Receiver Operator Characteristics) curve depicts the relationship between true positive rates (TPR) and false positive rates (FPR) across different classification thresholds. The given AUC values for negative, neutral, and positive classes are 57.41%, 55.01%, and 56.36%, respectively. Here's a brief interpretation of the resulting ROC curves:

- Negative class: The AUC of 57.41% indicates that the classifier has moderate performance when differentiating between negative samples and others. The ROC curve for the negative class would exhibit a gradual increase, reflecting the rising TPR as the FPR grows. However, due to the relatively low AUC, the curve wouldn't rise too sharply, indicating limited ability to separate negative samples from others.

- Neutral class: The AUC of 55.01% suggests that the classifier struggles to identify neutral samples. As a result, the ROC curve for the neutral class would be relatively flat, with little change in TPR as the FPR increases. This indicates that the classifier cannot effectively distinguish between neutral samples and other classes.

- Positive class: The AUC of 56.36% implies that the classifier performs slightly better than the neutral class when identifying positive samples. The ROC curve for this class would exhibit a mild upward trend, indicating that the TPR increases more rapidly than the FPR as the threshold is adjusted. However, the curve won't rise dramatically since the AUC remains below 58%.

In summary, the ROC curves for all three classes reveal that the classifier isn't performing optimally which is expected due to the miss-labeled dataset. The low AUC values and flat ROC curves for the neutral class suggest that improvements should focus on enhancing the classifier's capacity to recognize neutral samples. Additionally, further tuning the classifier's parameters and exploring alternative models might help boost performance for all classes.

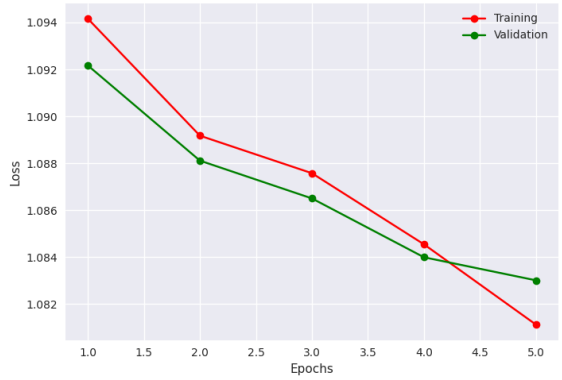

Figure 31: Learnig Curve (Loss)

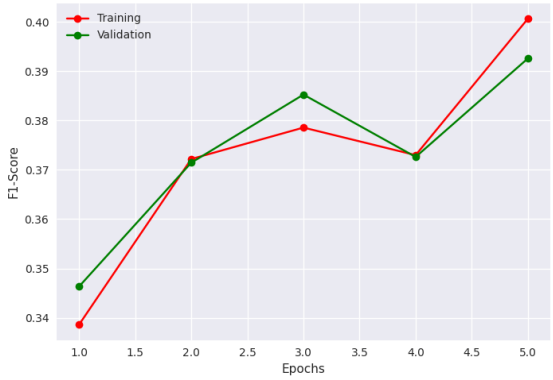

Figure 32: Learnig Curve (F1-Score)

According to the learning curve analysis, the model exhibits a steady improvement as the number of epochs increases. The reduction in both training and validation losses, as well as the increase in F1-score for both sets, indicates this. The learning curve for loss shows that both training and validation losses decrease as the number of epochs increases, indicating that the model is fitting the data better. The consistent decrease in validation loss confirms that the model is not overfitting. The learning curve for the F1-score also shows an increase in both training and validation sets as the number of epochs increases, indicating improved prediction accuracy. The simultaneous improvement in both training and validation sets suggests that the model is not prone to overfitting. Overall, the learning curves indicate that the model is well-designed and appropriately fine-tuned, and its development and resistance to overfitting are positive.

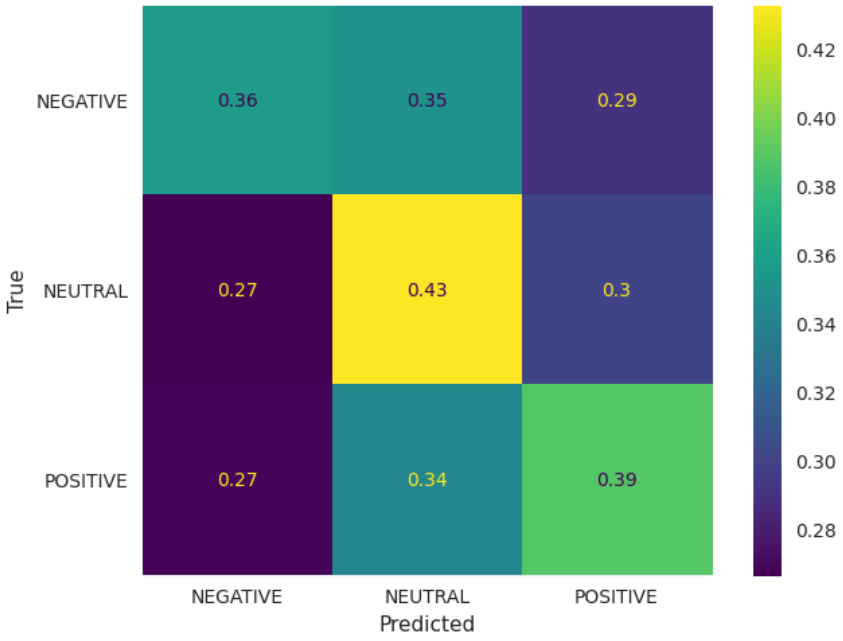

Figure 33: Final Model Confusion Matrix

A confusion matrix consists of the following:

- The true NEGATIVE is represented by the first row.
- The true NEUTRAL is represented by the second row.
- The true POSITIVE is represented by the third row.
- The predicted NEGATIVE is represented by the first column.
- The predicted NEUTRAL is represented by the second column.
- The predicted POSITIVE is represented by the third column.

The diagonal values are representing the correct prediction for each class:

- 36% for NEGATIVE
- 43% for NEUTRAL
- 39% for POSITIVE

6. Overall Analysis

6.1. Comparison with the first project

When comparing Logistic Regression with GreekBERT and DistilGREEK-BERT, it becomes apparent that the former lags significantly behind in terms of efficiency. GreekBERT exhibits over 600% slower performance, while DistilGREEK-BERT is approximately 300% slower, yet the improvements in metrics are only marginal. Additionally, Logistic Regression's advantage lies in its simplicity and rapid training, requiring mere seconds to develop. This simplicity translates into a more attractive option overall, as it demands considerably fewer computational resources and offers a more flexible modeling approach. Therefore, Logistic Regression emerges as the preferable choice in this comparison.

6.2. Comparison with the second project

When comparing the feed-forward neural network with BERT models, it becomes apparent that the former emerges as the superior choice. One notable advantage lies in its reliance on CPU instead of GPU, making it exceptionally appealing as not everyone has access to GPU resources, and online resources can be limited. Additionally, despite the availability of GPU resources, the improvements in metrics achieved by BERT models are only marginal. Given these factors, the feed-forward neural network presents itself as the preferable option.

6.3. Comparison with the third project

Among the projects that utilizes GPU, only the RNN and both BERT models stand out. While leveraging GPU resources, it's evident that both BERT models exhibit slow training times. Despite the minor improvements observed, the RNN's complexity becomes apparent with its requirement of configuring 10 hyperparameters. In my opinion, this complexity detracts from its attractiveness as an option. Consequently, I lean towards choosing one of the BERT models over the RNN due to the daunting task of tuning its parameters.

6.4. Comarison GreekBERT vs DistilGREEK-BERT

When comparing GreekBERT with DistilGREEK-BERT, the disparities are evident. GreekBERT demands nearly twice the training time of DistilGREEK-BERT for only marginal improvements in metrics. Moreover, our experiments revealed GreekBERT's susceptibility to overfitting. Considering these findings, it becomes clear that DistilGREEK-BERT emerges as the superior choice. Not only does it require fewer computational resources, but it also delivers commendable performance. In summary, DistilGREEK-BERT stands out as the more efficient and effective option.

7. Conclusion

Metrics show that the model is not as efficient as it could be; metrics are slightly above 40%. Despite that, the extensiveness of the procedure used lays a strong basis for further advancements. The constraints that have been found and the opportunities that remain untapped offer significant insights and may lead to improvements in future editions (personal projects).

8. Bibliography

References

[1] All TorchMetrics — PyTorch-Metrics 1.2.1 documentation.

[2] BERT Fine-Tuning Tutorial with PyTorch · Chris McCormick.

[3] How to Save a Plot to a File Using Matplotlib.

[4] optuna.visualization — Optuna 3.5.0 documentation.

[5] Saving and loading a general checkpoint in PyTorch — PyTorch Tutorials 2.2.1+cu121 documentation.

[6] sklearn.metrics.classification_report.

[7] Utilities for Tokenizers.

[8] nlpaueb/bert-base-greek-uncased-v1 at main, March 2022.

[9] EftychiaKarav/DistilGREEK-BERT at main, January 2024.

[10] Jason Brownlee. How to use Learning Curves to Diagnose Machine Learning Model Performance, February 2019.