Automated Essay Scoring: A Comparative Study

Senior Project

Primary Advisor: **Ali Faheem**
Secondary Advisor: **Dr. Muhammad Haroon Shakeel**

Presented by:

21-11482	Jam Ayub
21-11494	Muhammad Talha Imran
21-11386	Umama Rashid

Department of Computer Science

Forman Christian College (A Chartered University)

Automated Essay Scoring: A Comparative Study

By

Jam Ayub
M. Talha Imran
Umama Rashid

Project submitted to

Department of Computer Science,

Forman Christian College (A Chartered University),

Lahore, Pakistan.

in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE
IN
COMPUTER SCIENCE (Honors)

Ali Faheem — Primary Project Advisor

Dr. M. Haroon Shakeel — Secondary Project Advisor

Senior Project Management Committee Representative

Abstract

In today's world, when written and verbal communication is crucial, good writing abilities are vital. Essay writing is useful in the classroom as well as on several standardized assessments. Essay grading takes a lot of time and money. Because essay scoring has a high subjective component, it's difficult to utilize essay scores as an objective assessment criterion in standardized examinations. AES (Automated Essay Scoring) may be useful in resolving these issues. It is yet a very challenging problem of Natural Language Processing. The process includes essay length, grammar and spelling mistakes and many other components that affect the quality of essay. For that purpose, we studied and compared classical machine learning models, deep learning models and transformer-based NLP models to get best possible results for essay grading in terms of mean square error (MSE) and root mean square error (RMSE) score.

Acknowledgement

Alhamdulillah, all praise to Allah the Almighty for His countless blessings that has given us enough knowledge, the strength and courage to complete this project. We would like to offer our sincere gratitude to our advisors to encourage us and keep us all going throughout the whole journey; without their motivation and believing in us; this project would not have been completed. In addition, our greatest gratitude and appreciation to both of our advisors, Sir Ali Faheem and Dr. Muhammad Haroon Shakeel for their valuable guidance, advices, patience, and encouragement which made us complete our project in time. We are grateful for their expertise, experience and grasp over the domain of natural language processing and also how they provided us enough room to work in our own way. We would also like to thank all of the computer science department faculty at FCCU for the valuable knowledge they have given us during this journey. Finally, we expect that our project would be a beneficial source for anyone interested in this field of knowledge.

List of Figures

List of Tables

List of Equations

TABLE OF CONTENTS

Revision History

Name	Date	Reason For Changes	Version
Umama Rashid	1/10/21	Introduction revised	1.0
Talha Imran	18/10/21	Research methodology	1.1
Talha Imran	1/11/21	Requirement analysis	1.2
Jam Ayub	1/1/22	Results	1.3
Umama Rashid	1/5/22	Chapter 1 proof read	1.4
Talha Imran	29/01/22	Proof reading	1.5
Jam Ayub	30/01/22	Proof reading	1.6

Chapter 1. Introduction

1.1 Introduction

Manually grading essays takes a long time. Furthermore, human graders may be biased aimlessly during grading. It can result in ineffective grading and feedback that is inconsistent. On the other hand, using an unbiased training dataset can eliminate these drawbacks in the automated essay scoring system. As a result, the creation and use of mechanical essay grading systems are becoming more common.

Automated Essay Scoring is done by computers using grading models learned from essay datasets evaluated by various human graders. It is an application of Natural Language Processing (NLP) and a method of educational assessment. Cost, accountability, norms, and technology are all elements that contribute to the increased interest in automated essay scoring.

The urge to hold the educational system accountable for results by enforcing standards has grown due to rising school expenses. The advancement of information technology promises to minimize the cost of measuring academic success. Learning to write well is an essential part of secondary education, and it is a talent that can be improved with regular practice and formative feedback. Even the most dedicated teachers will find it difficult to provide targeted comments to every student on numerous revisions of each essay throughout the school year. Students can practice by taking tests and writing essays repeatedly to enhance the quality of their responses with automated essay grading.

Automated Essay Scoring (AES) uses statistical models for practical feature extraction from the essays and then assigns grades in a numeric range. It helps in reducing human efforts involved in the manual grading of papers and improves the efficacy and competency of writing assessments. Several models have been proposed for this purpose in the past few years.

An AES system's life cycle can be separated into two phases: the developmental phase and the operational phase. The AES system software is written during the development process, which can be a time-consuming task. The essays with accompanying scores, which are utilized as training data to develop a supervised machine learning classifier, are crucial for this step. The AES system will be fully operational when the development phase is

finished. The AES system is used to score new essays throughout the working period. The inputs are new essays with no scores, and the execution time is really brief. The AES system retrieves the relevant properties and assigns the appropriate scores to them.

1.2 Objectives

The following are the objectives of this study:

- We studied the artificial intelligence techniques that could be optimal for our problem.
- Study the machine learning models suitable for our research.
- It is finding the optimal model to score the essays.

1.3 Project Methodology

The section covers some of the neural network models and the motivation behind using these models in the project in order to produce the ideal comparative study on the dataset.

1.3.1 BERT

Bidirectional Encoder Representations from Transformers (BERT); an open-source machine learning framework for natural language processing (NLP). It's built on a deep learning model in which every output element is linked to every input element known as Transformer, and the weightings between them are determined dynamically based on their relationship. Language models were only able to interpret text input in one of two ways: right to left and vice versa, but not both at the same time. BERT is unique in that it can read in both directions simultaneously. The development of Transformers made this property of Bidirectionality possible.

BERT is pre-trained on two independent but related NLP tasks using this bidirectional capability: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). BERT is pre-trained on a plain text corpus which is unlabeled. Even when it is utilized in actual applications, it keeps learning unsupervised from the unlabeled text and never fails to improve. Its pre-training provides a foundation of information on which to develop. BERT may then fine tunes according to a user's demands and adjusts with respect to the growing corpus of searchable material and queries. Transfer learning is the term for this procedure.

It is made feasible by Google's Transformers research. BERT's greater potential for recognizing the meaning and ambiguity in language is due to the Transformer, which is a component of the model. Instead of processing each word individually, the Transformer processes a single word in connection to all other words in the phrase. The Transformer helps this model grasp the whole context of a term by keeping all the surrounding words in view, allowing it to better understand searcher intent.

1.3.2 ELECTRA

ELECTRA stands for Efficiently Learning an Encoder that Classifies Token Replacements Accurately. It is a novel pre-training strategy that seeks to meet or outdo the below average performance of an MLM pre-trained model while utilizing much less computational resources. In ELECTRA, the pre-training job is based on recognizing tokens that have been substituted into an input sequence. Two Transformer models are required for this arrangement such as a generator and a discriminator.

The steps in the pre-training process are as follows.

- Replace certain tokens with a token at random for a given input sequence. For all masked tokens, the generator predicts the original tokens.
- The discriminator's input sequence is constructed by substituting tokens with generator predictions. The discriminator predicts whether each token in the sequence is original or replaced by the generator.

The discriminator model is trained to identify which tokens have been replaced given a corrupted sequence, whereas for masked tokens the generator model is trained to predict the original tokens. While it conducts prediction on each token, the discriminator loss may be calculated over all input tokens. This loss is only calculated over the masked tokens when using MLM. This is demonstrated to be a significant difference between the two systems and the fundamental cause for ELECTRA's superior efficiency.

After pre-training, the discriminator model is employed for all the downstream tasks, while the generator is discarded. The loss of discriminator model is defined across all tokens in the sequence, as it must forecast whether each token is an original or not, as previously explained.

1.3.3 Naïve Bayes

Naive Bayes models are a group of extraordinarily fast and easy classification algorithms that are applicable to huge and massive datasets. They are extremely helpful as a fast baseline for a classification task considering they are quick and have a few configurable parameters.

Bayesian classification methods are used to create Naive Bayes classifiers. The Bayes theorem, which is an equation that describes the relationship between conditional probabilities of statistical data, is used in these methods. The probability of a label given in equation (1) describes observed features in Bayesian classification and can be defined as P(L | features). The Bayes theorem teaches us how to describe this in terms of more easily computed quantities:

$$P(L \mid features)=\frac{\mathbf{P(features \mid L)P(L)}}{\mathbf{P(features)}}$$

Equation 1: Bayes theorem

To differentiate and decide between two labels (L1 and L2), the ratio of the posterior from equation (2), shows probability for each label is one technique to make this conclusion.:

$$\frac{\mathbf{P(L1 \mid features)}}{\mathbf{P(L2 \mid features)}}=\frac{\mathbf{P(features \mid L1)P(L1)}}{\mathbf{P(features \mid L2)\,P(L2)}}$$

Equation 2: ratio of the posterior probability

The imaginary random process that generates the data, such a model is referred to as a generative model. The essential part of training a Bayesian classifier is to specify the respective generative model for each label. Although the general version of such a training step is a challenging operation, we can make it easier by making some simplifying assumptions about the model's shape.

There are several types of Naïve Bayes:

- Multinomial Naïve Bayes Classifier**:**

Feature vectors are used to represent the frequency through which a multinomial distribution produced specified events. For document classification it is the most common event model.

- Bernoulli Naïve Bayes Classifier**:**

Features are independent binary variables that describe inputs in the multivariate Bernoulli event model. This model is also popular for document classification problems, just like multinomial, where binary term occurrence characteristics are utilized instead of term frequencies.

- Gaussian Naïve Bayes Classifier:

In Gaussian Bayes continuous values related to every feature are expected to be distributed according to a normal distribution. It plots a curve that's parallel regarding the mean of the feature values once plotted.

1.3.4 Bi-LSTM

The LSTM network is a sort of RNN (Recurrent Neural Network) that is commonly used to solve problems involving sequential data prediction. LSTM, like any other neural network, contains layers that aid it in learning and recognizing patterns for improved performance. The basic operation of an LSTM can be seen of as holding the required data and discarding the data that isn't needed or beneficial for subsequent prediction.

There are many different forms of LSTM networks, however they can be loosely divided into three categories.

- LSTM forward pass
- LSTM backwards pass
- Bidirectional LSTM or Bi-LSTM

Bidirectional recurrent neural networks (RNN) are two separate Recurrent Neural Networks joined together. On every step, Bi-LSTM’s structure allows to have both forward as well as backward knowledge about the sequence at the same time.

Bidirectional will run the inputs in two directions, one from past to future and the other is vice versa. The feature that distinguishes this approach from unidirectional is that in the LSTM that runs backward, information from the future is preserved, whereas using these two

states combined, you can secure and save information from both future and past at any moment of time.

Bidirectional Recurrent Neural Networks (RNNs) have a simple concept. It demands the replication of the initial recurrent layer of the network so that there are two layers parallel to each other, then providing this input sequence to the first layer and a reversed duplicate copy of the input sequence to the second layer.

LSTM Recurrent Neural Networks have benefited greatly from this strategy. In the realm of speech recognition, the use of delivering the sequence bi-directionally was justified at first because there is evidence that rather than a linear interpretation, the context of the entire utterance is employed to interpret what is being said.

Bidirectional LSTMs may not be ideal for all sequence prediction issues, but they can provide some value in terms of better results in those domains where they are.

1.4 Problem Statement and Research Question

The following are the research questions for which this report seeks to provide the response to:

- How much performance gain can we achieve using new NLP architectures?
- What Machine Learning models can be used for automatic essay grading, and how have they implemented and yield results?
- Does new neural networks such as transformers performs better in our NLP problem?

Free form text and natural language used in daily human-to-human communication is an extensive and complicated problem to analyze for computer systems, whether in the form of a written sample or verbal interactions. The lack of one specific output of a communication task and obscurity in a given system language make grading and evaluating difficult. Generally, for this particular domain, the use of machine learning techniques on various features and large different patterned data sets can give us a lot of better results.

1.5 Scope

Teachers can use the AES system to grade students' essays. This system can also be used by testing agencies to test English content writing instead of hiring human graders. At the same time, the geographical scope of this system is universal. It is a vital machine

learning application that has been studied several times using different approaches and models. The future scope of this problem can be extended in many fields, such as searching and modeling fine syntactic and semantic features and devising an approach better than linear regression with neural networks.

Chapter 2. Requirements Analysis

In this section, we have discussed some deep learning models in recent years which have contributed mainly in automated essay scoring domain. Also, all the relevant information and some of the approaches in solving the problem is discussed. Furthermore, it explains the comprehensive review of the related work done on this domain.

2.1 Literature Review

Automated Essay Grading is an important educational application in the field of natural language processing. The study began in 1966 with the publication of Page's article [1], the forerunner of the Project Essay Grading System. The majority of individuals use a holistic approach to essay grading [2], [3], [4], [5]. This kind of grading assigns a single score to the essay. The reason for emphasizing holistic scoring is that corpora of manually annotated essay data sets are publicly available, and holistic scoring is more commercially valuable due to the automation of many essays written for tests like the SATs and GRE, which saves a lot of manual grading time.

Dimension	**Description**
Style	Word choice and structure of a sentence
Persuasiveness	The degree of presumably in argument
Relevance	Content relevancy
Coherence	Transitions of ideas
Cohesion	Appropriate use of phrases
Usage	Use of prepositions
Grammar	Grammar

Table 1: Dimension and Description of Text

From a research viewpoint, the most exciting aspect of the AES job is that it comprises a range of NLP issues of varying complexity. Table 1 provides the quality elements of the applicable scoring tasks to increase difficulty. For example, identifying grammatical and mechanical errors has been thoroughly explored and shown to be relatively successful. Finally, we have specific discourse-level issues involving the computer modeling of various characteristics of text structure like thesis clarity, persuasiveness, and coherence, which are understudied but undoubtedly tough. A few of these difficult aspects may need a

thorough concept of the given essay's subject, which is far apart from the capabilities today's essay scoring systems have.

Almost all AES systems on the market today are learning-based, and they may be classified as supervised, poorly supervised, or reinforcement learning. Researchers who use these supervised learning leads to AES have altered the task as:

- Regression task, in which the objective is to foresee an essay's score;
- Classification task, in which the objective is to categorize the essay into one of these classes (low, medium, or high, like they were in TOEFL11 corpus).
- Ranking task, in which the objective is to grade essays according to their quality.

Mass-produced learning techniques are often utilized to train models. The most often used regression techniques are linear regression, support vector regression, and sequential minimal optimization (SMO, a version of support vector machines). For categorization, SMO, logistic regression, and Bayesian network classification were used. Finally, SVM and LambdaMART were used to rank the candidates.

2.1.1 Regression Based Approach

The regression-based technique analyses feature values and essay score as independent and dependent variables, respectively, and then uses traditional regression algorithms, such as support vector regression, to develop a regression equation. On the request of the American College Board, Ellis Page created the first AES system, Project Essay Grader, in 1966. The PEG system employs a regression-based technique to estimate the score that human graders would give by defining a wide range of surface text elements from essays, such as the fourth root of essay length. E-rater is a commercial AES system created by Educational Testing Services (ETS) in America in the late 1990s and used in the Graduate Record Examination (GRE) and the Test of English as a Foreign Language (TOEFL).

2.1.2 Neural Approaches

Modern Automated Essay Scoring systems are mainly based on neural networks. During most of conventional AES research has centered on feature engineering, one of the most often mentioned benefits of neural techniques is, they do not need feature engineering.

First neural approach for holistic essay evaluation was presented by Ng and Taghipour in 2016 [5] . A convolution layer is used in their model to extract n-gram level properties from a succession of words in an essay. The features gathering the local textual relationships amidst the words in an n-gram, are further sent to a recurrent layer that is made up of an LSTM network [6], it then generates one vector each step of time that grasps the long-distance dependencies of the essay's words. After that, the vectors from various time steps are chained to create a vector which is put into a concentrated layer to foresee the grade of the respective essay. The above-mentioned one-hot input vectors are updated while the model is trained.

2.1.3 Word Embedding

Some terms aren't very effective at distinguishing between excellent and terrible writings. AES performance may be affected if these under informative words are not distinguished from their informative equivalents. Train word embeddings in light of this issue. Moreover, word embeddings are a low-dimensional actual value vector depiction of a word taught to locate two semantically comparable words in the word embedding space. "King" and "queen," for example, must have comparable embeddings, but "king" and "table" must not. As a result, word embeddings are widely thought to be a superior portrayal of word semantics than Taghipour and Ng's one-hot word vectors. However, word embeddings can also be trained on a huge, unlabeled data corpus using a word embedding learning neural network architecture that is also known as the CW model, propose enlarging the CW model with an extra output that corresponds to the essay grade in which the input word appears to train task specific word embeddings.

2.1.4 Using Attention

As previously said, certain characters, phrases, and sentences in an essay are more essential than others in terms of score and hence need greater attention. On the other hand, the two-convolution layer neural network by Dong [7] has failed to accomplish this. In order to detect key phrases, sentences and characters automatically, use attention pooling rather than basic pooling after each layer such as max or average pooling to add an attention mechanism into the network. The output of the associated convolution layer is taken as input by attention pooling layer and utilizes an amenable weight matrix in order to produce weighted combinations of the input vectors as output vectors.

2.1.5 Modeling Coherence

Because coherence is a key characteristic of essay quality, Cozma et. al. [8] predict that calculating and leveraging the coherence score of essays may enhance holistic grading. They demonstrate coherence in the following way. Like T&N, in place of their neural network they used LSTM. They also include a secondary layer in their neural network which accepts two LSTM positional outputs gathered at separate time steps as inputs and computes the resemblance between each set of positional outputs. The similarity values they got are known as neuronal coherence characteristics. The reason for this is because, on the surface, coherence and resemblance should be positively correlated. These characteristics are further employed to supplement the output vector of LSTM. Finally, they use the enhanced vector to forecast the holistic score, successfully using coherence in the scoring process. Learning Transfer In an ideal world, we'd be able to train prompt-specified AES systems in which the training and targeting prompts are similar, since it would enable AES systems to utilize the prompt-specific information they gained from training essays to score the testing essays more precisely. In reality, although, sufficient essays for goal questions are seldom accessible for instruction. In consequence, many AES systems are trained in a prompt-independent way, which means, only a few numbers of targeted prompt essays and a much greater number of non-targeted prompt essays are utilized for training. Anyhow, a possible dissimilarity in the language that is utilized in essays produced for the non-targeted prompt and those prepared for the targeted prompt might damage prompt-independent systems' performance. To overcome this problem, researchers used transfer learning (also known as domain adaptation) approaches to adapt the source prompts/domains to the target prompts/domains. EasyAdapt is a simple yet effective transfer learning technique that takes training data sets from just two domains (the source and the target) as input, to develop a model that can classify test examples taken by the destination domain effectively. In order to grasp EasyAdapt, a model without transfer learning that is often trained by using a feature space that is shared by instances from both the domains (i.e., source and destination domains). EasyAdapt enhances the respective feature set by replicating each feature in the space thrice, with the first copy stores the same information provided by both the domains, the second copy stores information from the source domain, and the third copy stores information from the target domain. It can be shown that the information from the target-domain will be assigned double the weight in this augmented feature space like the source-domain information, enabling the model to adapt to the target-domain information much better. Prompts may be seen as domains when applying transfer learning to AES. There is one goal prompt and many source prompts accessible for training in a realistic setting. Researchers that have used EasyAdapt

with AES consider every source prompt as belonging to the same source domain viewing the fact that EasyAdapt can only handle one from each target domain and source domain.

Existing techniques treat AES as a learning issue in general. Various learning approaches, such as classification, regression, and preference ranking, are used based on a large number of predetermined objectively quantifiable parameters [9]. The E-rater system uses natural language processing methods to extract several types of linguistic aspects from essays, such as lexical, syntactic, and grammatical features. The stepwise regression approach is then used to forecast the final score [10]. The classification-based technique treats essay scores as non-discriminative class labels, and predicts which class an essay belongs to using standard classification algorithms such as the K-nearest neighbor (KNN) and the naïve Bayesian model, where a class is connected with a numeric rating. The Intelligent Essay Assessor (IEA) [11] is a tool that assesses essays by evaluating semantic aspects. Each ungraded essay is scored based on the degree of similarity between its semantic vector and the semantic vectors of graded essays, as created using Latent Semantic Analysis (LSA) [12].

The above-mentioned study employs many sorts of characteristics, such as implicit and explicit features, to score the essay automatically using various models. The model's success is mostly determined by how well the retrieved feature describes the provided essay. Manually extracted features, word2vec feature representation, and embedding representation were the subject of certain studies. We think and describes in this paper that both manually extracted features and deep-encoded features might help AES models perform better. As a result, in this paper, we ran a large number of trials to assess word embedding in conjunction with manually derived minimal features and glove features while using different deep learning models.

2.1.6 Neural AES Models:

Alikaniotis 2016 [13] proposed a bidirectional Long Short-Term Memory (LSTM) deep neural network model that tapped into not just the context but also the use of words via a score-specific word embedding (SSWE). Alikaniotis tested a variety of neural network designs and found that the LSTM without the SSWE has been resulted in a considerable reduction in accuracy. Ng and Taghipour (2016) used an LSTM with a mean-over-time layer to beat the model performance of Alikaniosti in the same year, utilizing the same dataset, training methodology, and assessment measurement. Dong [7] employed a hierarchical two-layer Convolutional Neural Network (CNN) which is without word embeddings with levels

of accuracy equivalent to Alikaniost model due to the limitations of a stand-alone LSTM model in automated essay grading. Dasgupta et al. in 2018 [14] created a deep convolutional recurrent neural network for automated essay scoring that went farther than word and sentence embeddings that include highly developed psycholinguistic characteristics that seem to be inherent in a given text and reported levels of accuracy that outperformed the previously discussed neural network models.

2.1.7 LDA/LSA AES Models

On 283 essays with five distinct prompts [15] conducted a comparison research on essay grading utilizing Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA), and Probabilistic Latent Semantic Analysis (PLSA). They employed Spearman correlation to examine each method's inter-rater reliability. After doing the tests, they discovered that LSA and PLSA beat LDA, particularly on small test sets (100-150 essays). "Although LDA produced worse results, it has several theoretical benefits over the other two approaches (for example, dimensionality selection, being less prone to model overfitting, and the coherence of the generative model)". To adapt Latent Semantic Analysis (LSA) to Generalized Latent Semantic Analysis (GLSA), [16] substituted word by document matrix with n-gram by document matrix, which outperformed LSA. They experimented on writings ranging in length from 800 to 3,200 characters, and the results are comparable to human grades, with lower standard deviations than the LSA-based approach.

2.2 Design and Implementation Constraints

The main constraint of the project is the computer requirements for the models to run. As the models are very complex and large it requires enormous computation power required to run the models. Therefore, external cloud services are used to run the different models. For example, large transformers like BERT and GPT2 are very complex and large architectures and personal computers can not handle it.

2.3 Assumption and Dependencies

For creating the models for our task, we have used several libraries. Following are some libraries and tools which have been used in the project:

1. **Matplotlib:**

Matplotlib is a powerful python library that is known for its data visualization and plotting properties. Matplotlib is famous for creating different types of graphs and plots such as scatter plots, line plots, box plots, pie charts, histograms, bar charts, and many other visualization reports.

2. Pandas:

Pandas is another famous python package developed by Wes McKinney. It provides strong, flexible and expressive data structures, making data analysis and manipulation quite easy. The foundation of this library is Numpy, and Data-Frame is the vital data structure of Pandas, which allows the user to clean data, including removing missing values, row or column filtering upon some required criterion, and then saving tabular data back into rows and columns.

3. TensorFlow:

TensorFlow is an open-source plan used to create machine learning models. The Google Brain team of researchers basically created it to research ML and neural networks. TensorFlow APIs for python language are used by this project to train different models in order to achieve better results.

4. Nltk:

Natural Language Toolkit is a platform for developing python programs that are easy to work with natural language (i.e., human day to day language). Natural Language Processing (NLP) also written by NLTK creators supplies a brief introduction to program the natural language. Tokenization, word classification, stemming and many more interfaces are provided by NLTK which proved to be quite help in this project.

5. Sklearn:

Sklearn also known as scikit-learn is one of the useful libraries of python in the context of machine learning. It contains many statistical modelling tools which includes all the processes of regression, classification and clustering etc. This is

different from Pandas and Numpy as they are only confined to test summarization and manipulation while sklearn in majorly used to develop ML models.

6. Transformers:

Transformer's architecture is divided into encoder and decoder to generate an output instead of relying on recurrence and convolutions. The encoder maps the input sequence as a continuous representation taken as an input by the decoder. This encoder output along with the output of the decoder creates an output sequence as an actual output of the Transformer.

7. Google Collaboratory

The whole code for this project is written in Google Collaboratory, it allows the code writing and execution in the browser which makes it very easy to share between the members and even with the advisors. No configuration is required in Colab and it also provides free access to all GPUs. It also allows the coder to combine all of executable code, rich texts, images within a single document. On creating a Colab notebook, it is stored in the google drive account.

Chapter 3. System Design

This section defines the details of the dataset, tools, approaches, and techniques adopted to complete the project and our approach to compare different models in the thesis.

The dataset used in our project is "Automated Essay Prize (ASAP)". This dataset is sponsored by the William and Flora Hewlett Foundation [1]. There are eight sets of essays in this dataset and each set was generated from the single prompt.

Now the number of essays from each prompt varies from 900 to 1800, where the average length of each essay in terms of word count is 150 to 650. The dataset has total of 8 prompts/tasks

The essays that were written or this dataset were from varied classes and received a resolved score – the actual grade; from human professionals. The dataset also comes with the validation dataset that can be used for fine tuning, the most important thing is that the validation dataset does not overlap with the training dataset. However, we have not used the separate validation dataset.

The prompts are divided into two types of writing styles:

- Persuasive/Narrative/Expository
- Source Dependent

The Persuasive/Narrative/Expository type gathers the taps the student's general declarative knowledge while the source dependent essays focus more on students' domain-specific knowledge in literature and physics. Or social studies. Each prompt varied in the degree of knowledge that was required. Two professional graders holistically graded the prompts. Also, the essay's data were also anonymized using a Named entity Recognizer (NER) from the standard NLP group.

[1] https://www.kaggle.com/c/asap-aes/data

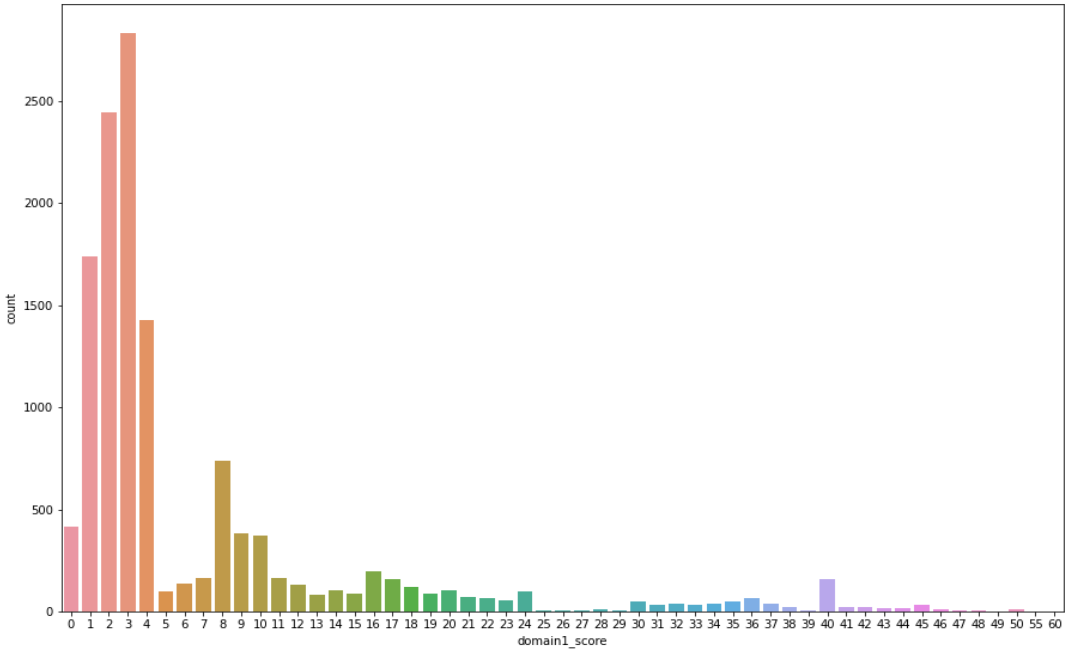

Figure 1: Count of Essay Scores in dataset

Figure (1) shows the count of scores in the whole dataset from prompt 1 to prompt 8. We can see that the score is skewed left, and most of the score which is given to essays is from 1-4. The reason behind the score is from prompt 1-6; the score is given from 0-12. However, from the 7-8 prompt, the score was given from 0-60 while the only 1 essay got 60 marks.

3.1 Normalization

Normalization is the process of transforming features into a consistent scale. This enhances the model's performance and training stability. Suppose that from equation 3, we have a data set of X; in our case essays; have N rows (essay_id, essay_set, etc.) and D columns (domain1_score). *X [:,i]* represent the feature *i* and *X [j,:]* represents entry j then the equation will be

$$\hat{X}[:,i] = \frac{X[:,i] - min(X[:,i])}{max(X[:,i]) - min(X[:,i])}$$

Equation 3: min max normalization

While this will rescale all the values in range of [0,1]. While the most standard format used in essay grading is from scale of 0-10 or 0-100. So, all the values which were rescaled in range of [0,1] were multiplied by 10 in order to get a 0-10 scale.

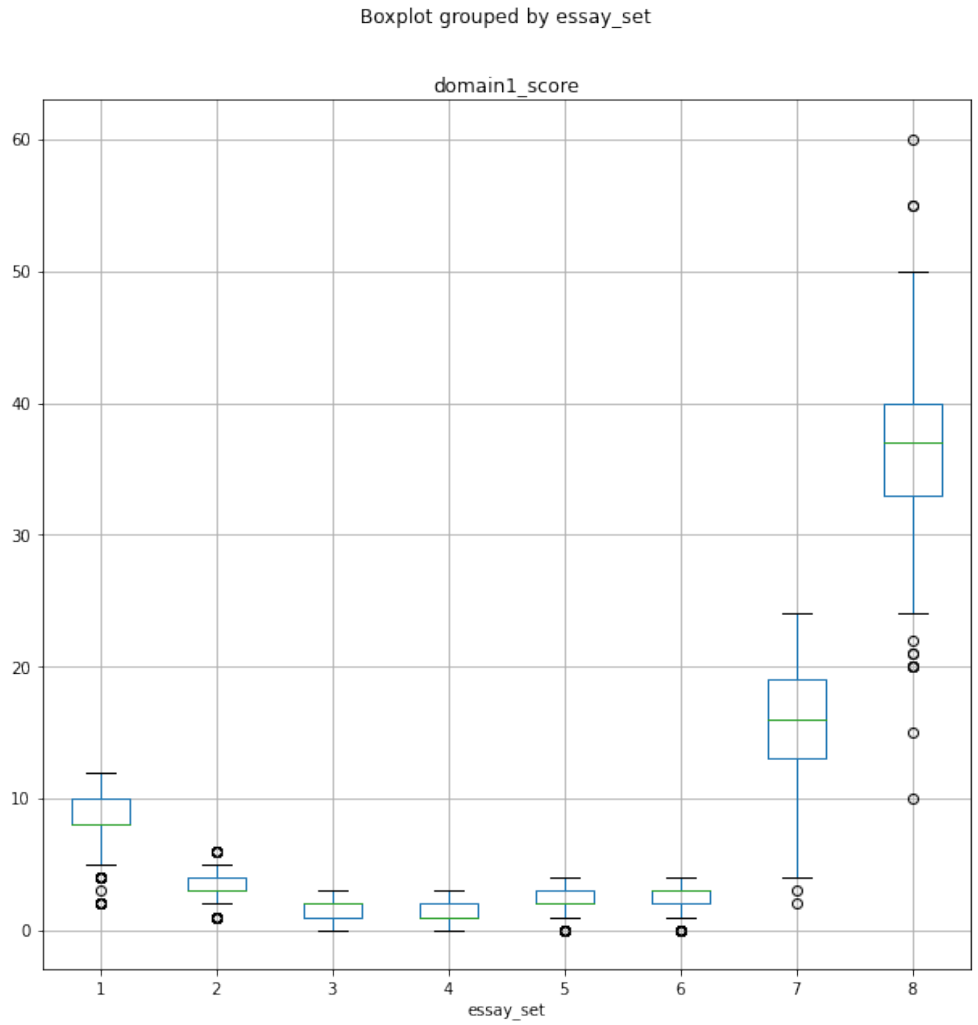

Figure 2: original score distribution in essays

Figure. (2) shows that more refined presentation of the dataset score. From the Fig. (4), we can observe that prompt 1 has a mean score ranging from 8-10 while some population had below the score of 5 while none had a score of 0. Moving forward to prompt 2 the upper cap was approximately near the prompt 1 threshold values while the lower cap is equivalent to the minimal value of prompt 1.

Prompts 3 and 4 have the same grading score from 0 to 8 while the prompts 5 and 6 present a score somewhat higher than the prompts 3 and 4.

Prompt 7 and 8 were scored from 0 to 60 range and the upper cap of the prompt 7 is equal to the lower cap of prompt 8 score i.e. 25. The prompt 7 students got score ranging from 2 to 25 while the mean is 13 to 18. The mean of prompt 8 is from 32 to 40 range and 1 was scored 60 as well as 1 score of around 55 can be observed. The smallest value in prompt is at 10.

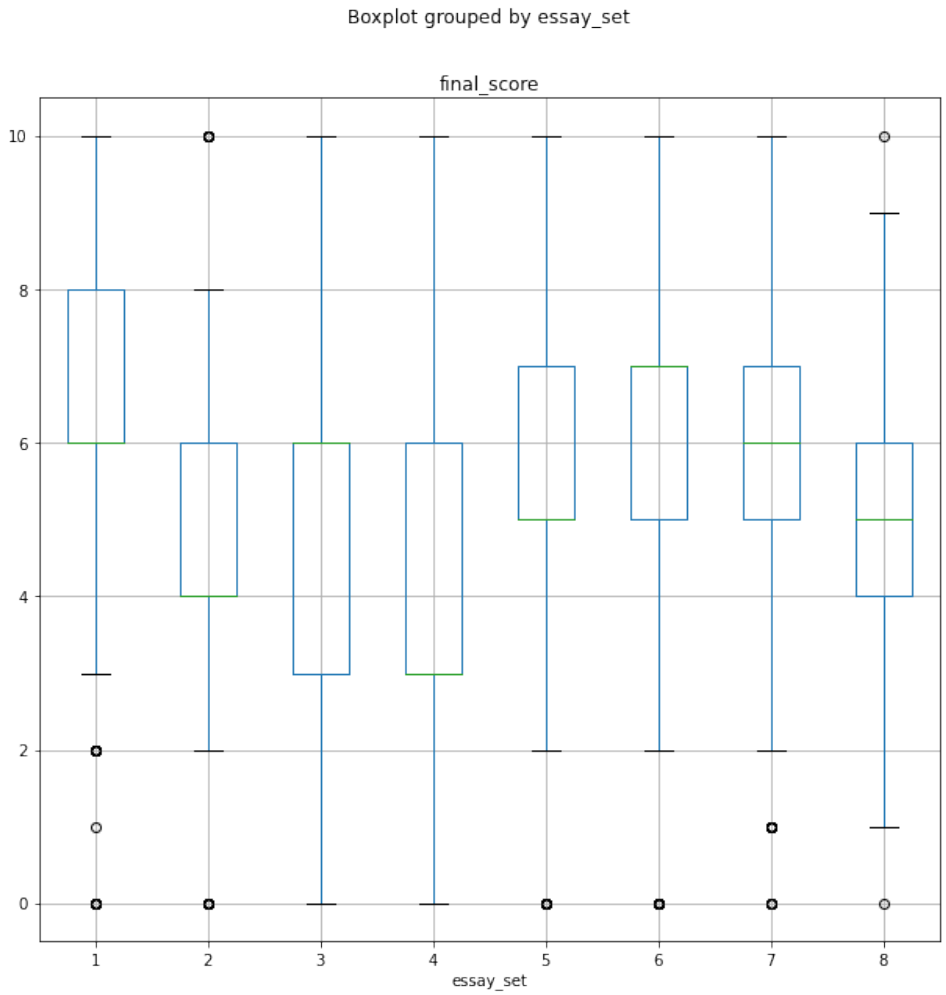

Figure 3: score distribution after normalization

Figure (3) shows that after normalization the score of essay set 1 is ranged from 0 to 10 while the upper and lower are 10 and 3 respectively few scores are present in ranges of 0-2 which is presented in circle. Likewise, essay set 2 scores ranges between 2 to 8 while few values can be seen on score 0 and 10 and the most important thing is that no values are observed on score 1 in this essay set. Essay set 3 and 4 covers the full range of score from 0-10 while moving forward to set 5 and 6 similar patterns is observed, the score spread is same and set 7 has score spread from 2 to 10 while fewer values present on 0 and 1 score. essay set 8 has some values on 0 and 10 while the spread is between 1 and 9.

3.2 Data Division

The data division is 60% for training while the 20% is validation and 20% is for testing.

3.3 Pre-processing

First stage in the pre-processing is removing the blank rows if any; present in the dataset. After that the text was changed to lower case and then tokenization was done. Tokenization is a common NLP task. It separates the piece of text into smaller units known as token. Token can be words, characters, sub words.

After that pos tagging was done. Part-Of-Speech (POS) tagging is a special label which is assigned to each token in a text corpus to indicate the part of speech. Only three pos tagging was done namely – verb, adjective and adverb.

After POS tagging lemmatization was done, which is a crucial part of the linguistic problem. it has the purpose of reducing a word's inflectional form and oftentimes derivationally related form to a general basis.

For example:

am, is, are → be

the result of this will be in mapping of text like:

the girl's dolls are different color variation → the girl's doll be differ color variety

Also, we have used the stop words which is the list of the common words which does not provide the useful information in the text in most of the text analysis procedure. Some of the words from stops words are "a", "of", "an", etc.

3.3.1 Word Embeddings

The word embedding techniques are used to represent words mathematically. There are many embeddings namely Term Frequency – Inverse Document Frequency (TF-IDF), Word2Vec, FastText etc.

We've employed TF-IDF in traditional classifiers, which is a statistical metric for determining the mathematical importance of words in texts. One Hot Encoding is comparable to the vectorization technique. Instead of 1, a TF-IDF value is given to the value corresponding to the word. Multiplying the TF and IDF numbers yields the TF-IDF value.

While in neural networks we have used Word2Vec, The whole corpus is scanned, and the vector construction procedure is carried out by calculating which words contain the target word most often.

In case of Transformers text embeddings are done by the transforms its self.

3.4 Architecture of various models

In this section, the loss functions of the models and the architecture representation of neural networks are described.

3.4.1 Loss function

The loss function used in the classifiers, neural networks and transformers is calculated on different functions which are described as follows:

3.4.2 Mean Square Error

Mean Squared Error (MSE) is the simplest and most common loss function is world of machine learning. The calculation of MSE is straightforward – by taking the difference between the model's prediction and the true score of the essays and then by taking square and taking average it throughout the dataset. Also, the value of MSE can never be negative due the fact that the value gets squared.

From equation (4), It is defined as the N number of scores in essays and y is the predicted score of essays and $\hat{y}$ is the true label or score in the essays.

$$\boldsymbol{MSE = \frac{1}{N}\sum_{i=1}^{N}(yi - \widehat{yi})^2}$$

Equation 4: MSE Formula

3.4.3 Cohen's Kappa Score

The Cohen's Kappa Score *K* is a metric that is used to assess agreement between the two raters. The score is the inter-rater agreement between two graders. Assume that we an essay

and ask two raters to give score to an essay in range of [0-10] p_e, then the score can be measured by the level of agreement between those two raters through kappa score as shown in equation (5).

$$K = \frac{p_0 - p_e}{1 - p_e}$$

Equation 5: Cohen's Kappa Score

3.4.4 Traditional classifiers

As a part of comparative study, we have conducted an experiment of traditional classifiers namely Multinomial Naïve Bayes, Logistic Regression, XG Boost Classifier.

The data was fed after pre-processing and a classification report with confusion matrix was generated. The MSE and Root Mean Squared Error (RMSE) with Kappa Score was generated.

3.4.5 Neural networks

Neural networks are computational models made up of linked nodes that function similarly to neurons in the brain. They can discover hidden patterns and correlations in raw data using algorithms, cluster and categories it, then learn and improve over time.

3.4.5.1 Activation Function

An activation function is used neural networks in order to help the network in learn the complex patterns in the data. Usually the function is added at the end of the neuron and it calculated what to fire to the next neuron in the model. the functions also add the non-linearity into the output of the neurons

In our models we have used rectified linear activation function or ReLU which outputs the input directly if it is positive. The activation considers two values and output the maximum from both values.

The activation function was also adopted in transformers.

3.4.5.2 Optimizer

An optimizer is very important in the neural networks. it is a function or an algorithm which can change the attributes in the neural network like the neuron weights and learning

rates in the model. Eventually, it helps to reduce the overall loss and aimed at improving the accuracy of the model.

In our project we have used the Adam optimizer. Adaptive Moment Estimated or Adam is an algorithm the used for optimization of gradient descent. This is very efficient when working with large dataset or involving of multiple parameters. By considering the 'exponentially weighted average' of the gradients, this approach is utilized to speed up the gradient descent process. Using averages accelerates the algorithm's convergence to the minima. The same optimizer was used in transformers.

3.4.5.3 Model Architecture

The essays were first standardized and vectorized by the embedding layer of glove 300d. the vectorized text was sequentially passed to the fully connected layer with filter size of 31 which was passed to the max pooling layer with pooling size of 2. 2 fully connected layers of LSTM were added with 100 and 64 nodes respectively. The second layer of LSTM with 64 nodes has recurrent drop out of 40%. We have trained the layers with batch size of 128 and learning rate of $6^{-5}$with 500 epoch and early stopping. Then after both layers a single dropout layer we have used linear activation function in the last layer. For loss function mean square error was used with rmsprop optimizer the model is describes in figure (4).

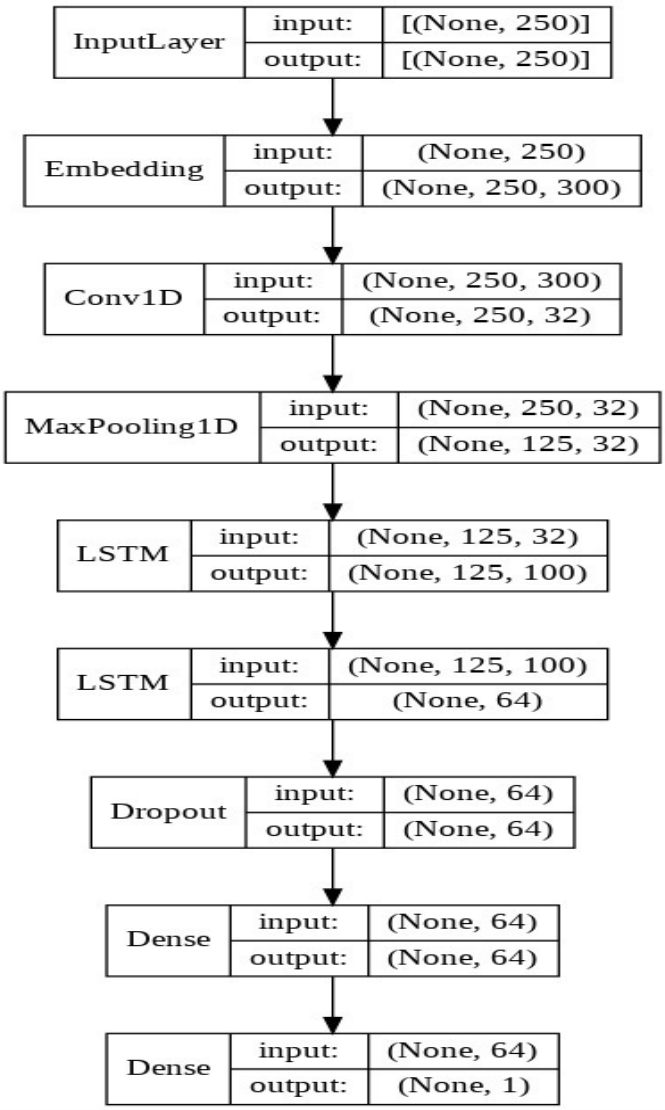

Figure 4: Sequential Model of CNN+LSTM

The model is a sequential CNN and Bi LSTM network. The input essays were first tokenized by GloVe embedding (pre-trained embedding) with 300 dimensions. The vectorized text was fed to the CNN convolutional layer of 250x32. We have trained the layers with batch size of 128 and learning rate of 6^{-5}. After that max pooling was used with stride of size 2. After that dense layer with 64 nodes and 64 dimensions with L2 regularization 0.02 was implemented. The result was dense layer was forwarded to LSTM layer of 100 nodes and recurrent drop out of 0.2. Our Bi-directional LSTM layer has 64 nodes and recurrent drop out of 0.2 and epoch was 500 with early stopping. our activation function was linear and loss function was mean square error and choose adam optimizer the model is shown in figure (5).

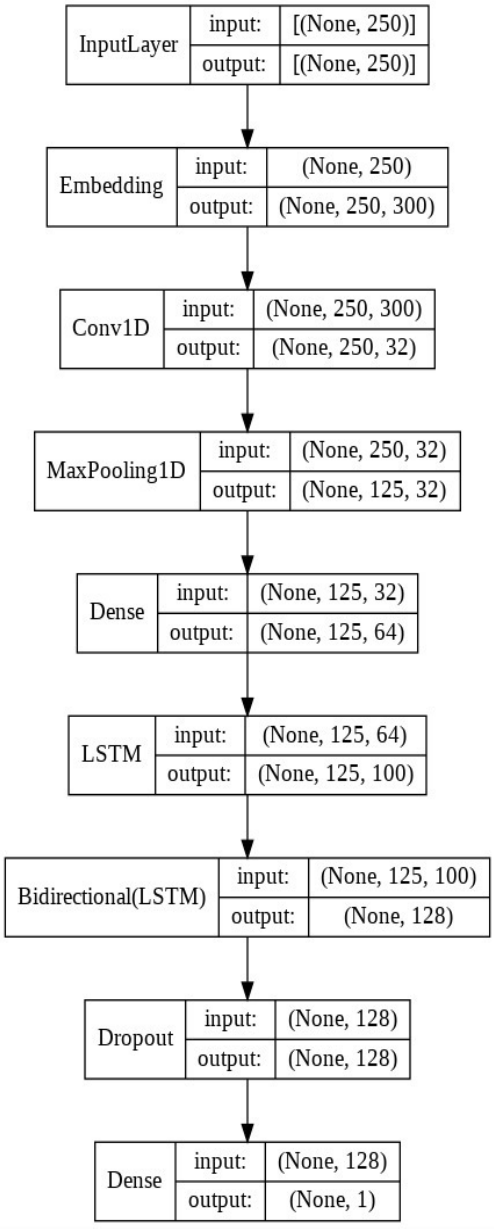

Figure 5: Sequential Model of CNN+BiLSTM (baseline model)

In the functional API the tokenization process same as of previous two but this time the result was forwarded to three parallel fully connected layers as shown in in figure (6). The two layers were CNN convolutional layers with 64 filter and stride size of 3 and 2 respectively. We have trained the layers with batch size of 128 and learning rate of 6^{-5}. The third output of embedding was passed to Bi LSTM of 50 nodes. The result of three layers were concatenated and passed to the dense layer of 64 nodes; a regularization of l1 was implemented in this layer and the result was fed to another dense layer with activation function ReLU. We have used ReLU activation in 2 more layers after that with nodes of 32 and 16 respectively after that the output layer has 1 node and the activation function was linear. Our loss function was a mean square error while optimizer was adam.

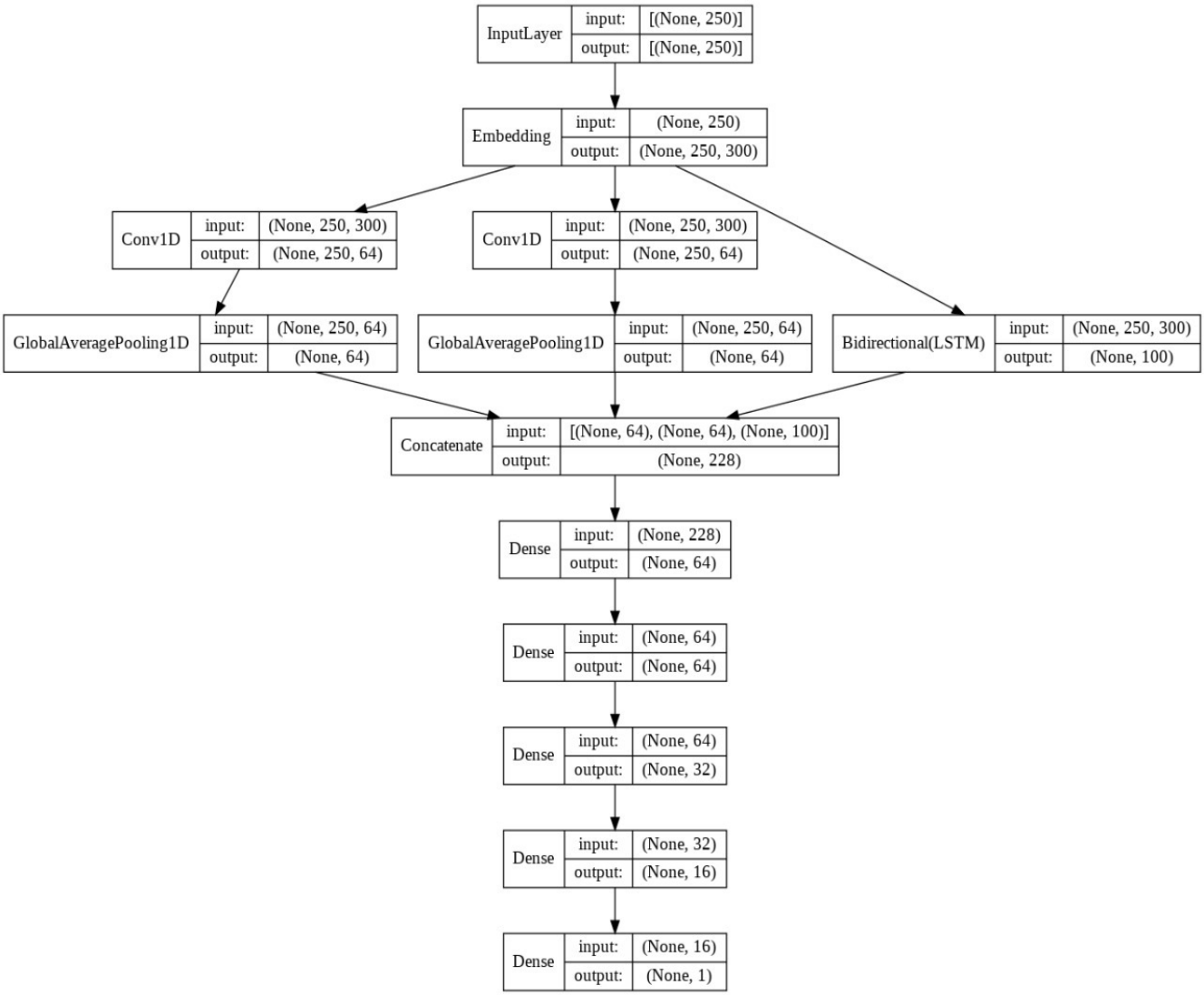

Figure 6: Functional API of CNN+BiLSTM

3.4.6 Transformers

Before learning about transformers, it is necessary to learn about the Sequence-to-Sequence (Seq2Seq) models in NLP that converts sequence type A to sequence type B. the example of this problem is translation of languages e.g. translating Urdu to English or vice versa.

Like from the name of Seq2Seq, it prevents the parallelization in task. The transformers addressed this challenge.

Transformer is a novel and complex architecture the solve this problem and proposed first time in paper Attention Is All You Need *[17]*. It adopts on the mechanism of self-attention and weighting the significance of each part of the input data. Transformers can handle sequential input but it does not process data in order. The Transformer does not every time process the sentence in a sequential manner like the end of sentence is processing at the end, rather in a contextual meaning of each word in the sentence which allows the parallelization.

3.4.6.1 Model Architectures

We experimented with a pre-trained transformer-based BERT model as shown in figure (7) followed by three fully connected layers for the easy scoring. We applied the BERT tokenizer. The intermediate layers of convolution were implemented. Our batch size was 128 and learning rate of 6^{-5}. The activation was ReLU in both of convolutional layers. The output of bert embedding was fed to Bi-LSTM. the result of other two convolutional layers and BI-LSTM was concatenated forwarded to dense layer of 64 nodes. After three more dense layer the result was fed to linear function layer.

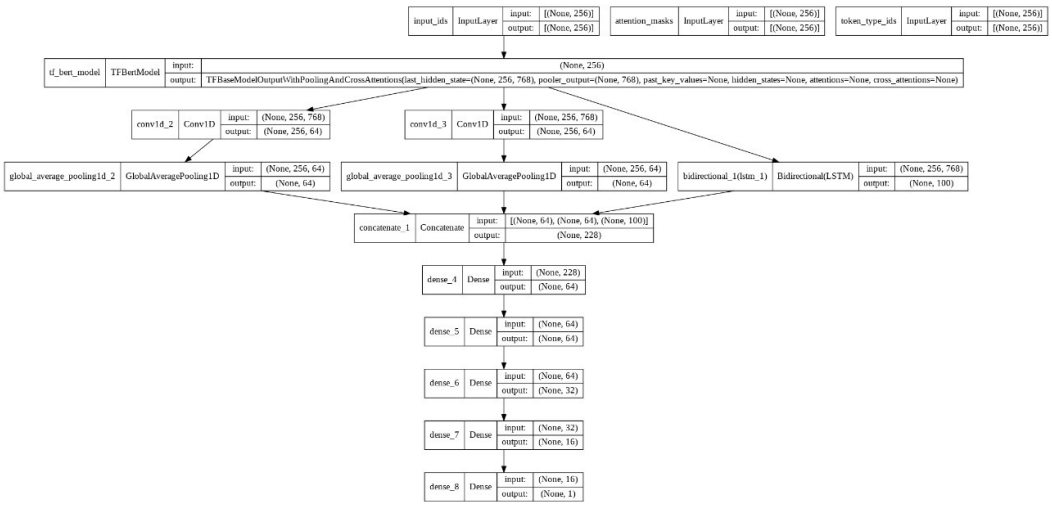

Figure 7: BERT base uncased

We have also experimented with pre-trained transformer-based RoBERTa model as shown in figure (8), the stretgy was same as previous models the batch size and learning rate were constant Our optimizer was adam while the loss function is mean square error. In convolutional layers the stride size was 2.

Figure 8: RoBERTa base

Chapter 4. Results

This section covers the results (through experimenting different models) which were generated after feeding the data to models. These models have been trained on data after normalizing the it. By the help of different graphs and tables the results are described.

4.1 Experiments

The library which was used in plotting graphs and classification reports are sklearn and seaborn. In terms of neural network training we have used callback function that can conduct action at different stages of learning like stopping the training at certain number of epochs when the model calculates the same mean square error after certain epochs.

In terms of classifier and neural network, the results are presented in confusion matrix and mean square error.

In addition to that, the learning curves are also presented for neural networks.

Moreover, the classification report and confusion matrix were generated throughout the project from classifiers to neural networks. A classification report is used for measuring the quality of prediction. This indicates that how many predictions are True Positive, True Negative, False Positive and False Negative. The report consists of four key values:

- Precision
- Recall
- F1-score
- Support

Precision refers to the ability of model not labelling an instance positive that is actually negative as stated in in equation (6). It is defined as the ratio of true positive to sum of true and false positives.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FP})}$$

Equation 6: Precision

The second important point of classification report if recall which presents the percentage of positive cases the model can identify as shown in equation (7). To conclude, it is the ratio of true positive to the sum of true positive and false negative.

$$\boldsymbol{Recall = \frac{TP}{(TP + FN)}}$$

Equation 7: Recall

The third point is the F1 score, which tells about the correct positive prediction. The weighted harmonic mean of precision and recall and the range of score falls in [0.0-1.0]. it is observed that the F1 score is generally lower than the accuracy measure, considering precision and recall to calculation. Moreover, the intuition behind the F1 score is to compare the models not the accuracy. The equation (8) describes the F1 score as:

$$\boldsymbol{F1\ Score = 2 * \frac{(Recall * Precision)}{(Recall + Precision)}}$$

Equation 8: F1 Score

Support shows the actual occurrences of the class in the specific dataset. It tells about the true response that falls under the class.

The results of the different models can be comprehended based on the classification report, confusion matrix of the models.

4.1.1 Trail 1: Traditional Classifiers

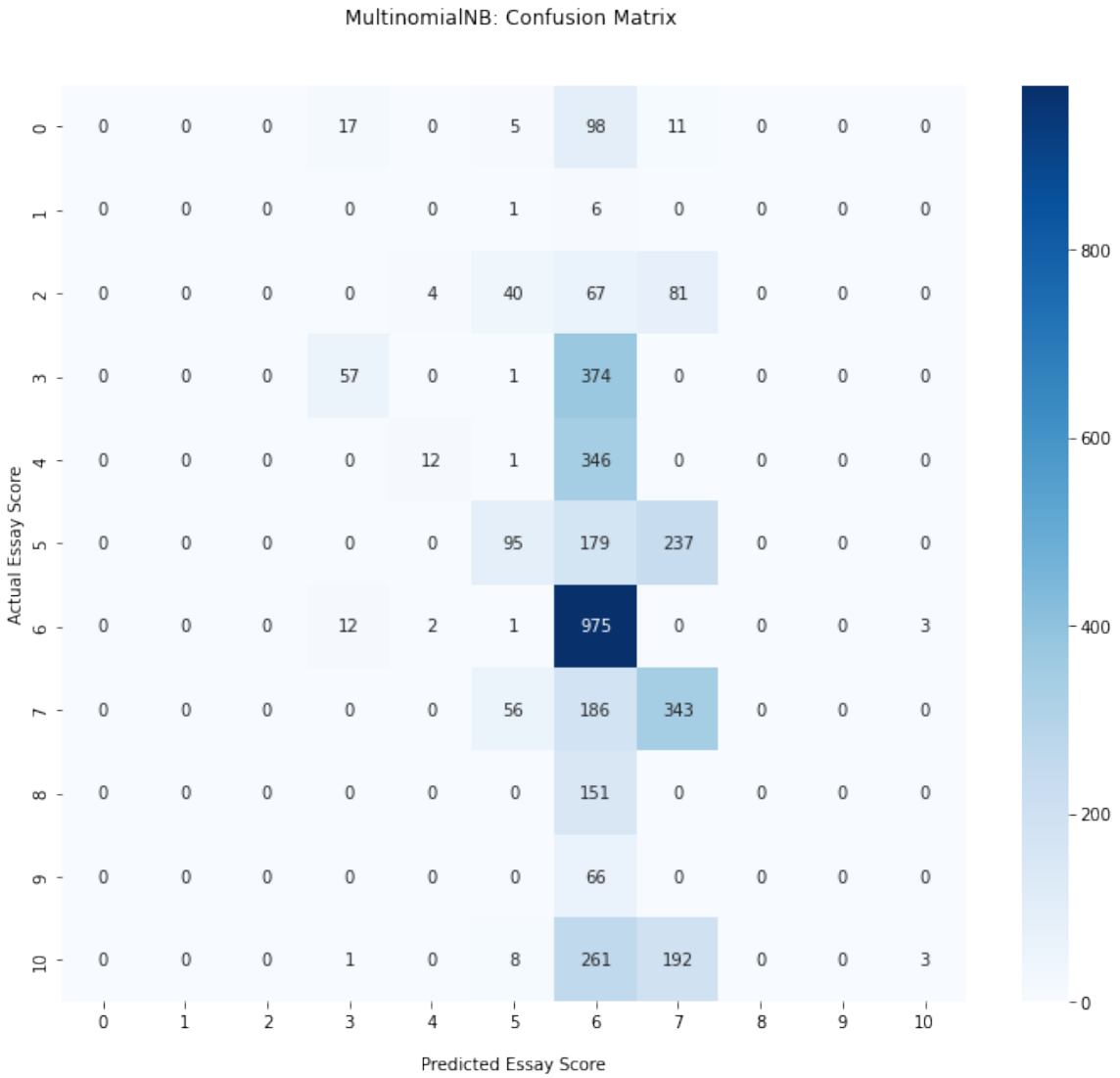

Figure 9: Confusion Matrix of Multinomial Naïve Bayes

	precision	recall	f1-score	support
0	0	0	0	139
1	0	0	0	11
2	0	0	0	209
3	0.641026	0.120773	0.203252	414
4	0.333333	0.005208	0.010256	384
5	0.485437	0.213675	0.296736	468
6	0.358198	0.988787	0.525888	981

7	0.426637	0.62069	0.505686	609
8	0	0	0	161
9	0	0	0	55
10	0.375	0.006508	0.012793	461
accuracy	0.386177	0.386177	0.386177	0.386177
macro avg	0.238148	0.177786	0.141328	3892
weighted avg	0.360909	0.386177	0.271509	3892

Table 2: Classification report of Multinomial Naïve Bayes

The confusion matrix and the classification report of the model showing the results of classes from 0 to 10 in figure (9). The matrix is showing the most of the values in predicted category from 5-7 or the mean of the score. Multinomial NB has predicted large number of values accurately with the score of 6, however the score is not correctly spread due to the fact the there is no representation of other score like 0,1,2, and 8,9 in the confusing matrix. The report shows the values of support in classes of 0,1,2 but the model has not predicted the classes very well which represent the boundaries of the dataset. Similarly, the trend can be seen from class 8 and 9 where the model has not predicted the values very well. From the f1 score from table (2) we can conclude that the model has able to predict class 6 very well because it has the highest value in among the classes. Half of the prediction in class 6 are not true as we can see from the recall and the f1 score.

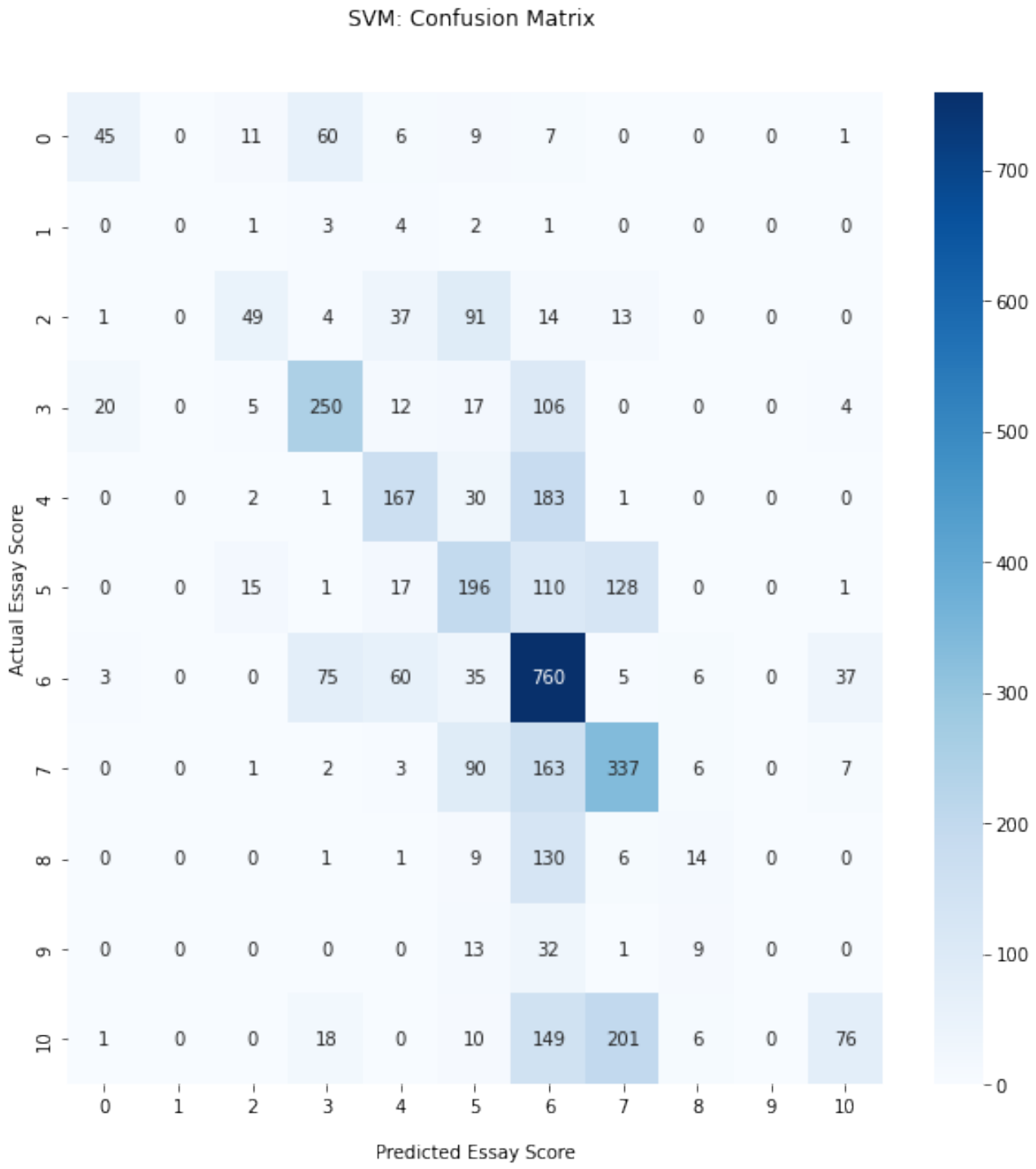

Figure 10: Confusion Matrix of SVM

	precision	recall	f1-score	support
0	0.642857	0.323741	0.430622	139
1	0	0	0	11
2	0.583333	0.23445	0.334471	209
3	0.60241	0.603865	0.603136	414
4	0.543974	0.434896	0.483357	384
5	0.390438	0.418803	0.404124	468
6	0.459215	0.77472	0.576631	981
7	0.486994	0.553366	0.518063	609
8	0.341463	0.086957	0.138614	161
9	0	0	0	55
10	0.603175	0.164859	0.258944	461
accuracy	0.486639	0.486639	0.486639	0.486639
macro avg	0.423078	0.326878	0.340724	3892
weighted avg	0.496503	0.486639	0.456594	3892

Table 3: Classification report of SVM

The confusion matrix and the classification report of the model from figure (10) showing the results of classes from 0 to 10. The matrix is showing the most of the values in predicted category from 2-7. SVM has predicted large number of values accurately with the score of ranging from 2 to 7. However, there was not score observed in class of 1 and 9.

The highest f1 score in report from table (3) has class 3 which is predicted very much correctly following class 6 and 7. The model able to predict these scores very accurately. While the class 0 and the 10 which are the boundaries has the highest precision in the report. The model was able to predict the score of class 3, 5 and 7 correctly as their value of recall and f1 score is near to similar.

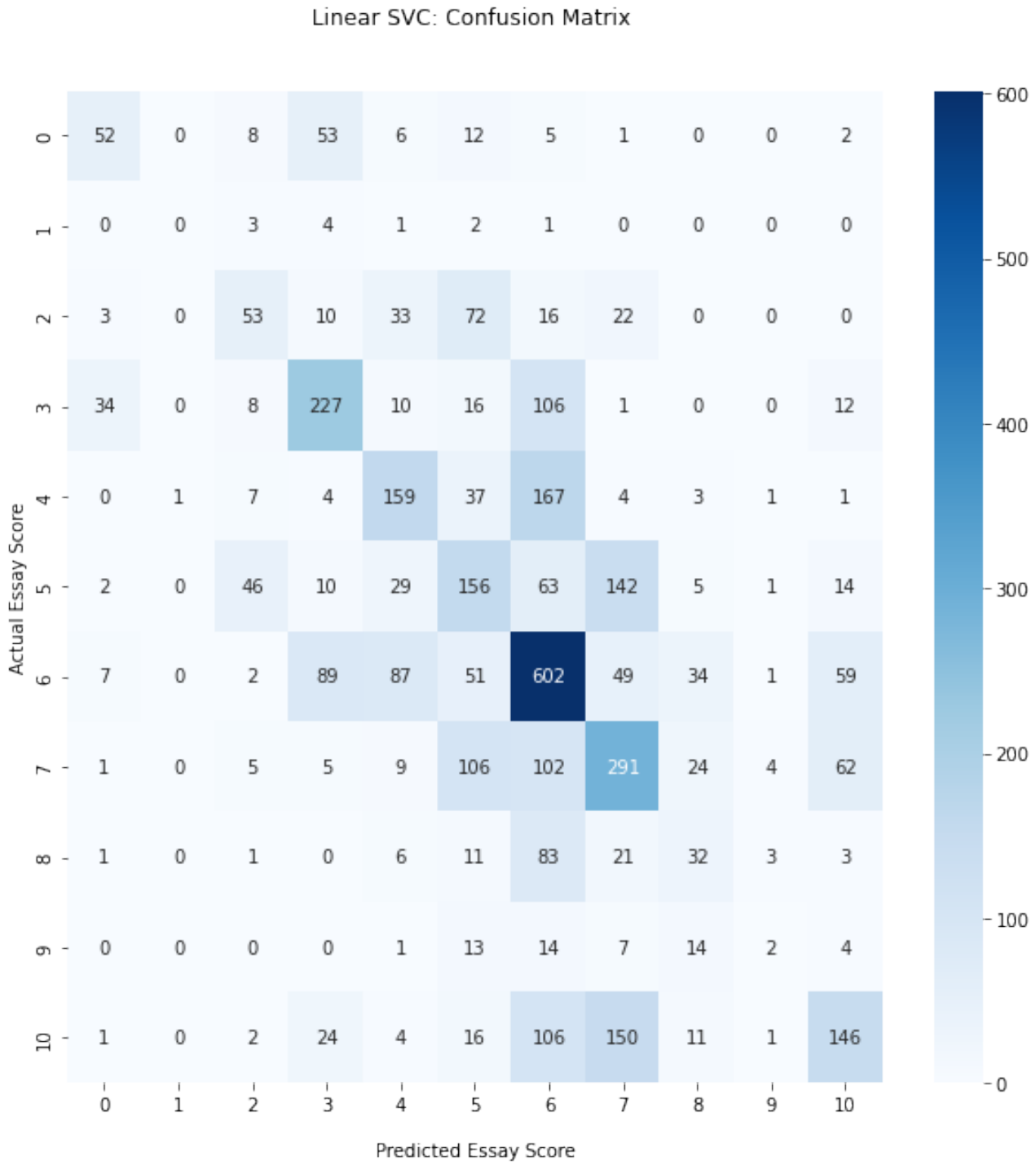

Figure 11: Confusion Matrix of Linear SVC

	precision	recall	f1-score	support
0	0.514851	0.374101	0.433333	139
1	0	0	0	11
2	0.392593	0.253589	0.30814	209

3	0.532864	0.548309	0.540476	414
4	0.46087	0.414063	0.436214	384
5	0.317073	0.333333	0.325	468
6	0.475889	0.61366	0.536064	981
7	0.422965	0.477833	0.448728	609
8	0.260163	0.198758	0.225352	161
9	0.153846	0.036364	0.058824	55
10	0.481848	0.316703	0.382199	461
accuracy	0.441932	0.441932	0.441932	0.441932
macro avg	0.364815	0.324246	0.335848	3892
weighted avg	0.435894	0.441932	0.43239	3892

Table 4: Classification report of Linear SVC

The confusion matrix and the classification report of the model showing from figure (11) the results of classes from 0 to 10. The matrix is showing the most of the values in predicted category from 2-7. Linear SVC has predicted large number of values accurately with the score of ranging from 2 to 7. However, there was not score observed in class of 1. Unlike SVM linear SVC has predicted almost every class but the score of class 1 is not observed. The model has the highest recall in class 6 while has the lowest of 0 in terms of class 1. The model has the very poor precision in identifying the classes of 1, 8 and 9. The weighted average of precision and recall from table (4) is 0.43 and 0.44 respectively.

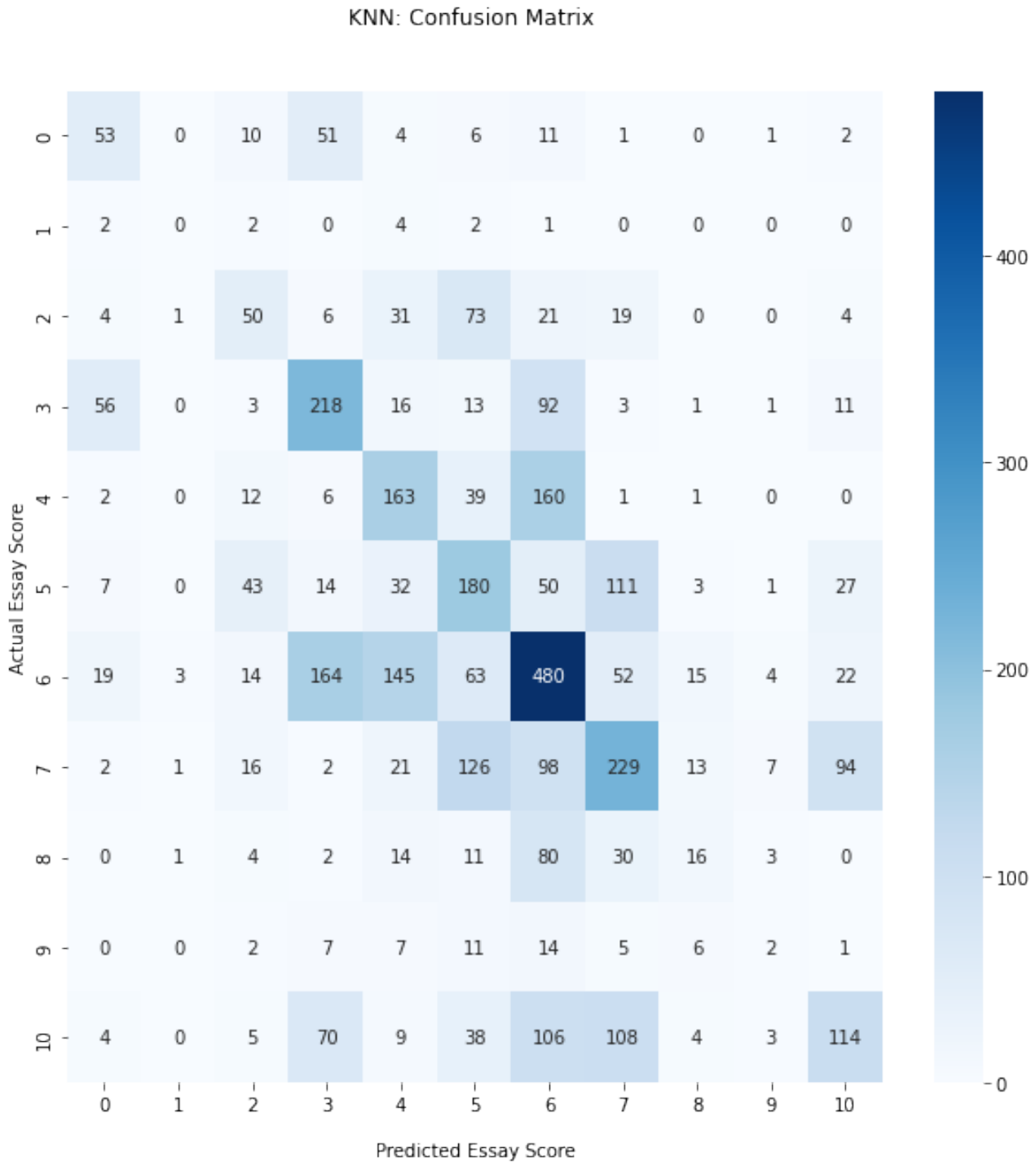

Figure 12: Confusion Matrix of KNN

	precision	recall	f1-score	support
0	0.355705	0.381295	0.368056	139
1	0	0	0	11
2	0.310559	0.239234	0.27027	209

3	0.403704	0.52657	0.457023	414
4	0.365471	0.424479	0.392771	384
5	0.320285	0.384615	0.349515	468
6	0.431267	0.489297	0.458453	981
7	0.40966	0.376026	0.392123	609
8	0.271186	0.099379	0.145455	161
9	0.090909	0.036364	0.051948	55
10	0.414545	0.247289	0.309783	461
accuracy	0.386691	0.386691	0.386691	0.386691
macro avg	0.306663	0.291323	0.290491	3892
weighted avg	0.381305	0.386691	0.37741	3892

Table 5: Classification report of KNN

The confusion matrix and the classification report of the model showing from figure (12) the results of classes from 0 to 10. The model was able to predict the values ranging from 0 to 10, unlike previous models KNN was able to predict the score in class of 1 but the score was correct. As we can see that the actual score of class 1 is 0 while the cross value of class 1 is also 0 but the score from other actual classes was assigned with score 1. If we see the repot of the model we came to know that the precision and recall has worst in terms of score 1. The recall for score 6 was higher than any other score in the model. the precision was somewhat similar in all score space. The weighted average of precision and recall from table (5) is 0.38 and 0.38 respectively.

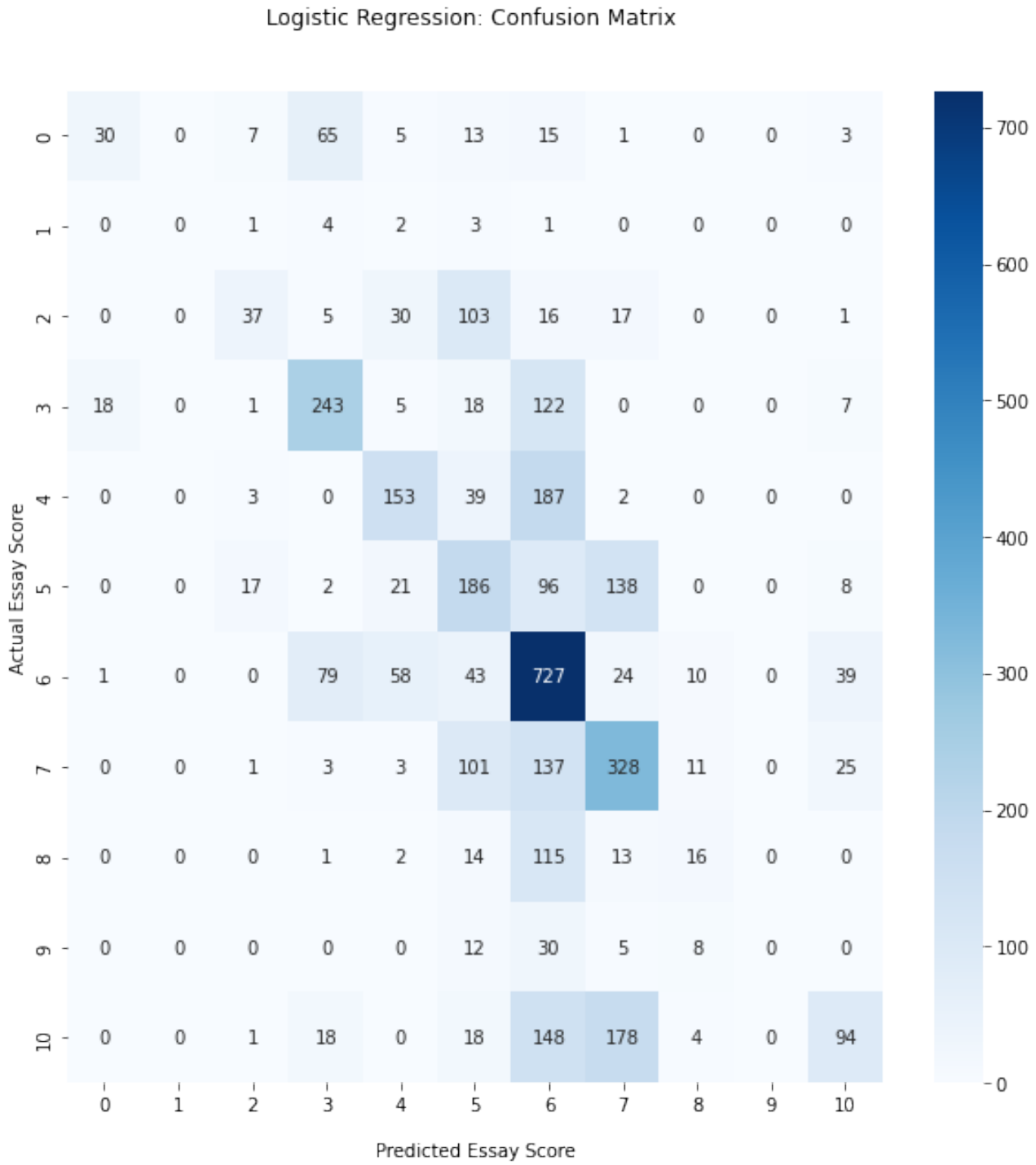

Figure 13: Confusion Matrix of Logistic Regression

	precision	recall	f1-score	support
0	0.612245	0.215827	0.319149	139
1	0	0	0	11
2	0.544118	0.177033	0.267148	209

3	0.578571	0.586957	0.582734	414
4	0.548387	0.398438	0.461538	384
5	0.338182	0.397436	0.365422	468
6	0.456085	0.741081	0.56466	981
7	0.464589	0.538588	0.498859	609
8	0.326531	0.099379	0.152381	161
9	0	0	0	55
10	0.531073	0.203905	0.294671	461
accuracy	0.466084	0.466084	0.466084	0.466084
macro avg	0.39998	0.305331	0.318778	3892
weighted avg	0.471468	0.466084	0.4388	3892

Table 6: Classification report of Logistic Regression

The confusion matrix and the classification report of the model showing from the figure (13), the results of classes from 0 to 10. The model was able to predict the values ranging from 0 to 10, the recall of all the score were somewhat similar except score 9 and 1 which the model was not able to predict. The model has the highest precision in terms of score 0. While the highest recall can be observed in score 6. The weighted average of precision and recall from table (6) is 0.47 and 0.46 respectively.

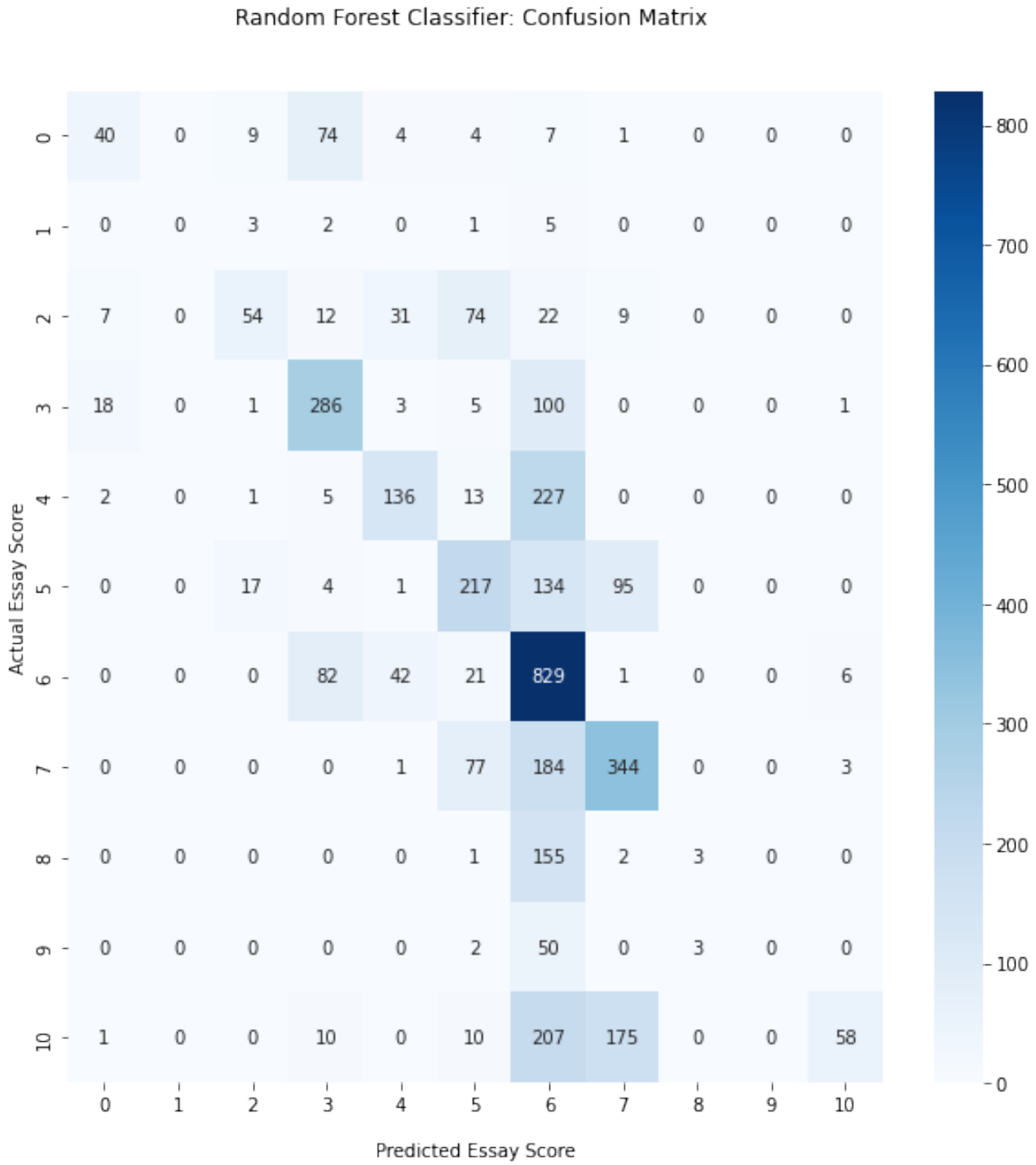

Figure 14: Confusion Matrix of Random Forest Classifier

	precision	recall	f1-score	support
0	0.588235	0.28777	0.386473	139
1	0	0	0	11

2	0.635294	0.258373	0.367347	209
3	0.602105	0.690821	0.64342	414
4	0.623853	0.354167	0.451827	384
5	0.510588	0.463675	0.486002	468
6	0.431771	0.845056	0.571527	981
7	0.548644	0.56486	0.556634	609
8	0.5	0.018634	0.035928	161
9	0	0	0	55
10	0.852941	0.125813	0.219282	461
accuracy	0.505396	0.505396	0.505396	0.505396
macro avg	0.481221	0.328106	0.33804	3892
weighted avg	0.558511	0.505396	0.463606	3892

Table 7: Classification report of Random Forest Classifier

The confusion matrix and the classification report of the model showing from the figure (14), the results of classes from 0 to 10. The model was able to predict the values ranging from 0 to 10, the recall of all the score were somewhat similar except score 9 and 1 which the model was not able to predict. The model has the highest precision in terms of score 10. While the highest recall can be observed in score 6. All of the score lies in the similar space of precision value expect score 1 which has 0. The weighted average of precision and recall from table (7) is 0.55 and 0.50 respectively.

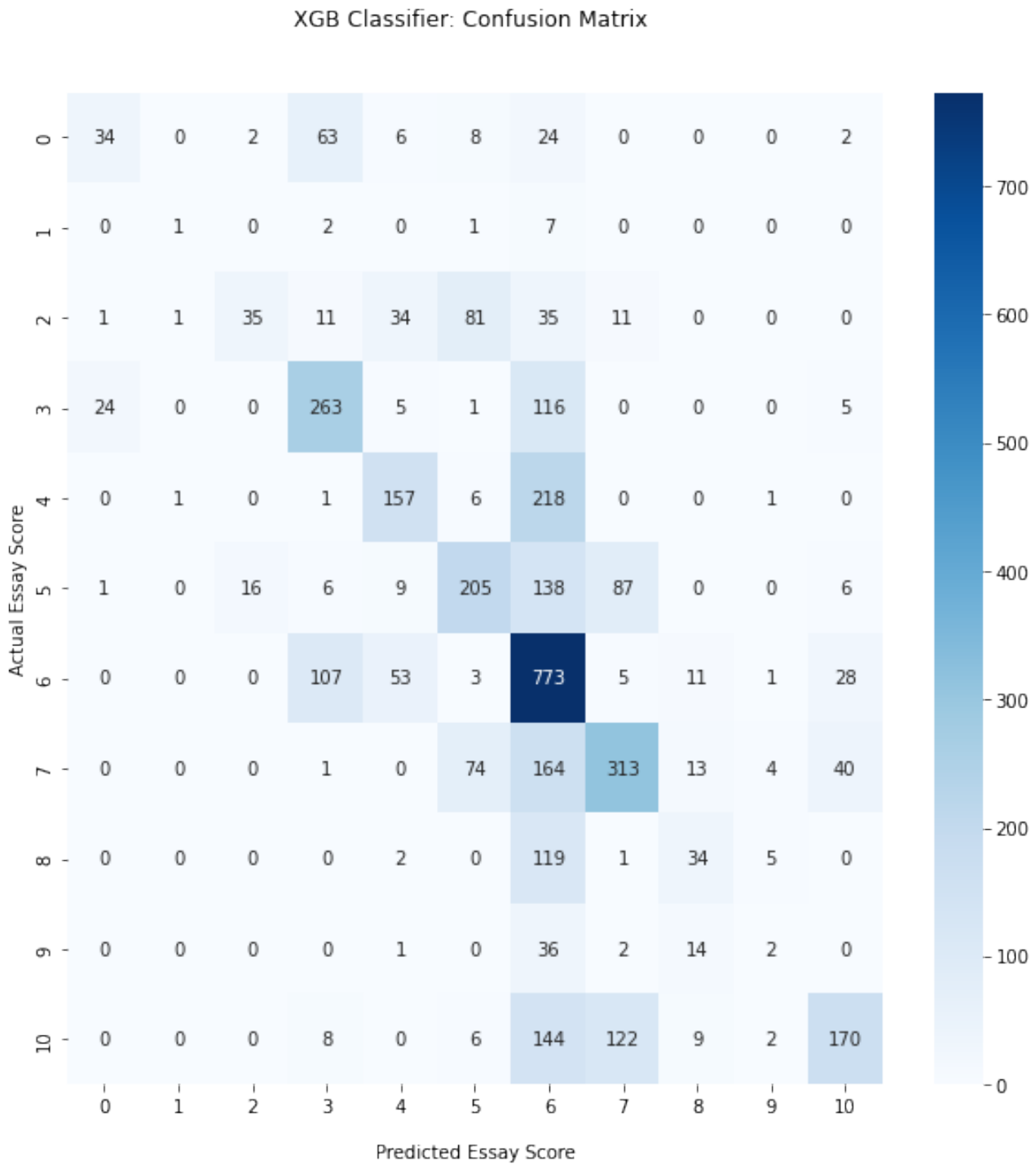

Figure 15: Confusion Matrix of XGB Classifier

	precision	recall	f1-score	support
0	0.566667	0.244604	0.341709	139
1	0.333333	0.090909	0.142857	11
2	0.660377	0.167464	0.267176	209

3	0.569264	0.635266	0.600457	414
4	0.588015	0.408854	0.482335	384
5	0.532468	0.438034	0.480657	468
6	0.435738	0.787971	0.561162	981
7	0.578558	0.513957	0.544348	609
8	0.419753	0.21118	0.280992	161
9	0.133333	0.036364	0.057143	55
10	0.677291	0.368764	0.477528	461
accuracy	0.510534	0.510534	0.510534	0.510534
macro avg	0.499527	0.354852	0.385124	3892
weighted avg	0.539071	0.510534	0.491827	3892

Table 8: Classification report of XGB Classifier

The confusion matrix and the classification report of the model showing from the figure (15), the results of classes from 0 to 10. The model was able to predict the values ranging from 0 to 10, the recall of all the score were somewhat similar except score 9 and 1 which the model was not able to predict. The model has the highest precision in terms of score 10. While the highest recall can be observed in score 6.

The highest deviation of precision can be observed in score 9. The weighted average of precision and recall from table (8) is 0.53 and 0.51 respectively.

4.1.2 Trial 2: Neural Networks

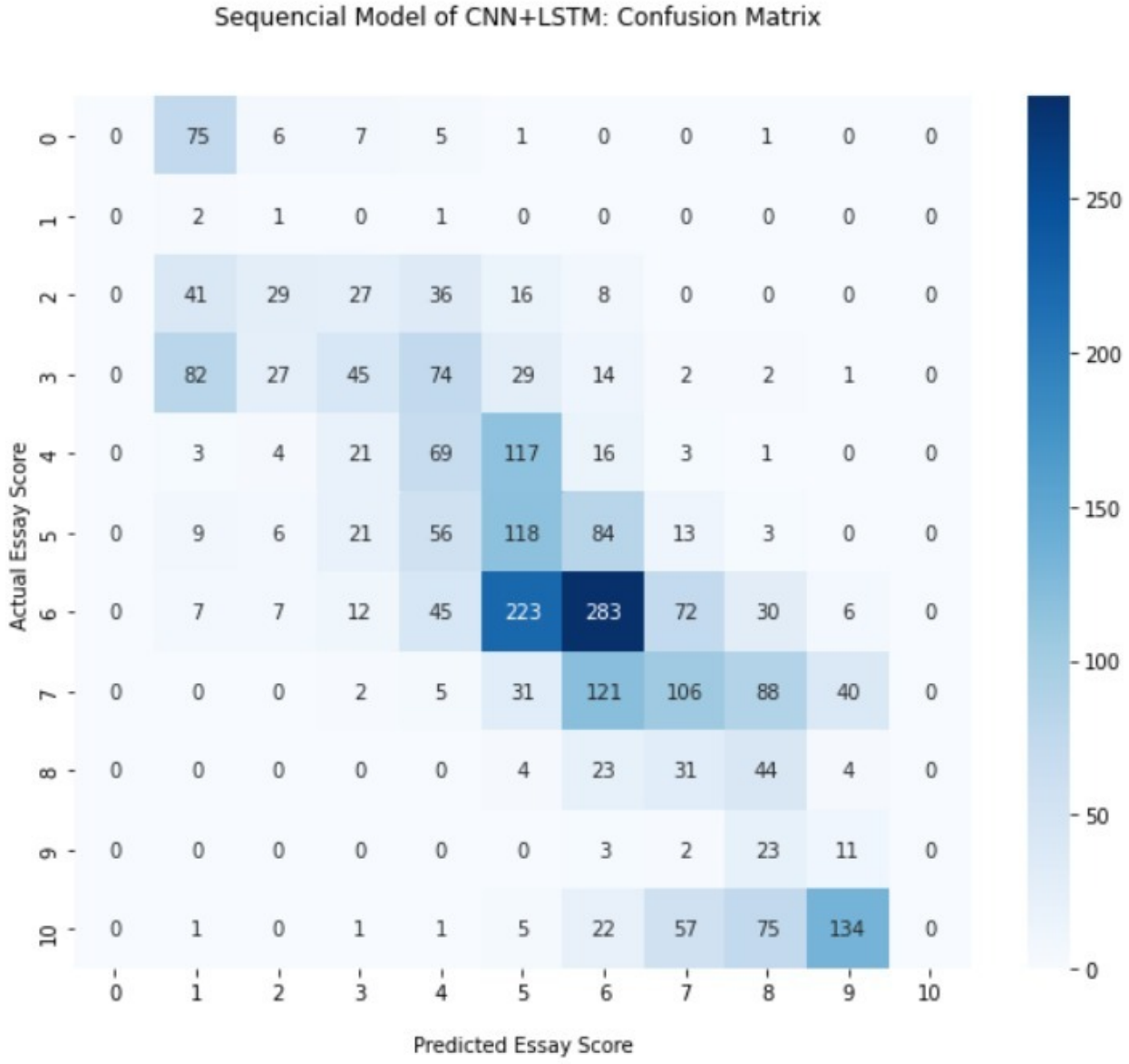

Figure 16: CNN+LSTM Confusion Matrix

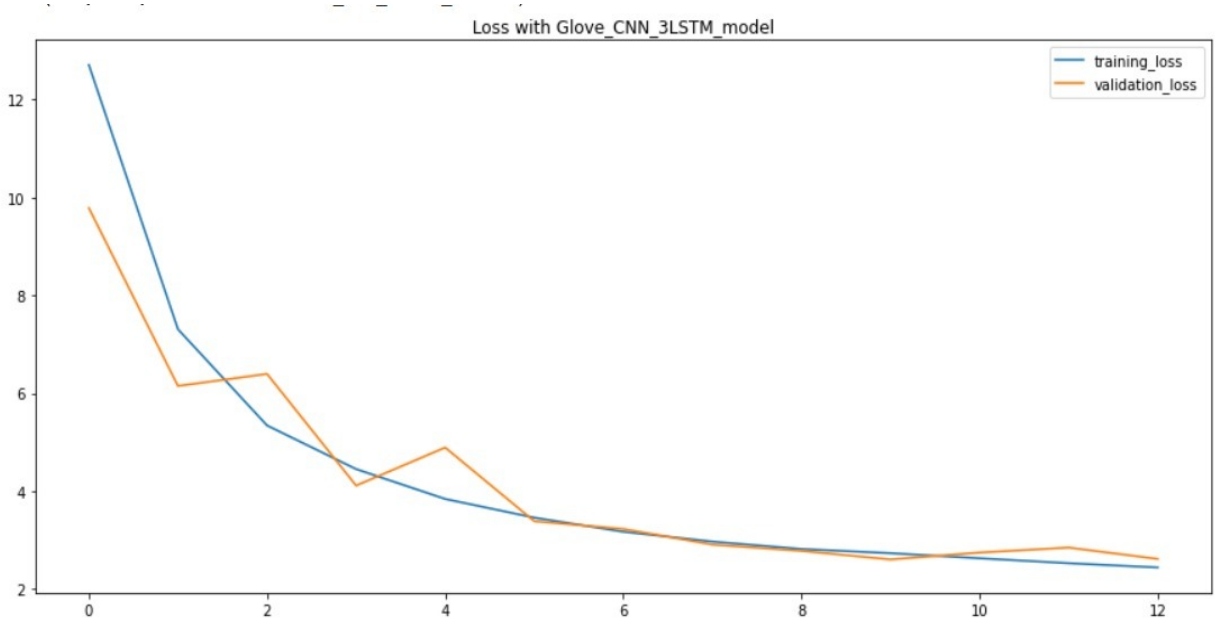

Figure 17: Learning Curve of CNN+LSTM

	precision	recall	f1-score	support
0	0	0	0	95
1	0	0	0	4
2	0.153846154	0.063694268	0.09009009	157
3	0.328042328	0.224637681	0.266666667	276
4	0.246527778	0.303418803	0.272030651	234
5	0.205714286	0.348387097	0.258682635	310
6	0.481949458	0.389781022	0.430992736	685
7	0.33423913	0.312977099	0.32325887	393
8	0.121212121	0.339622642	0.17866005	106
9	0.056666667	0.435897436	0.100294985	39
10	0	0	0	296
accuracy	0.26743738	0.26743738	0.26743738	0.26743738
macro avg	0.17529072	0.219856004	0.174606971	2595
weighted avg	0.274644431	0.26743738	0.260775015	2595

Table 9: Classification report of CNN+LSTM

For the neural networks we have presented the confusion matrix in figure (16), learning curve in figure (17) and the classification report of the model in table (9). The learning curve is

generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has not predicted the score 10 and 0 to the essays. The actual 0 score was predicted as 1 and 10 was predicted 9. It can also be seen from the matrix that upper bracket of predicted score was mostly correspond to the range of 7 to 10.

The weighted precision of the model is 0.274644431 while the recall is 0.26743738

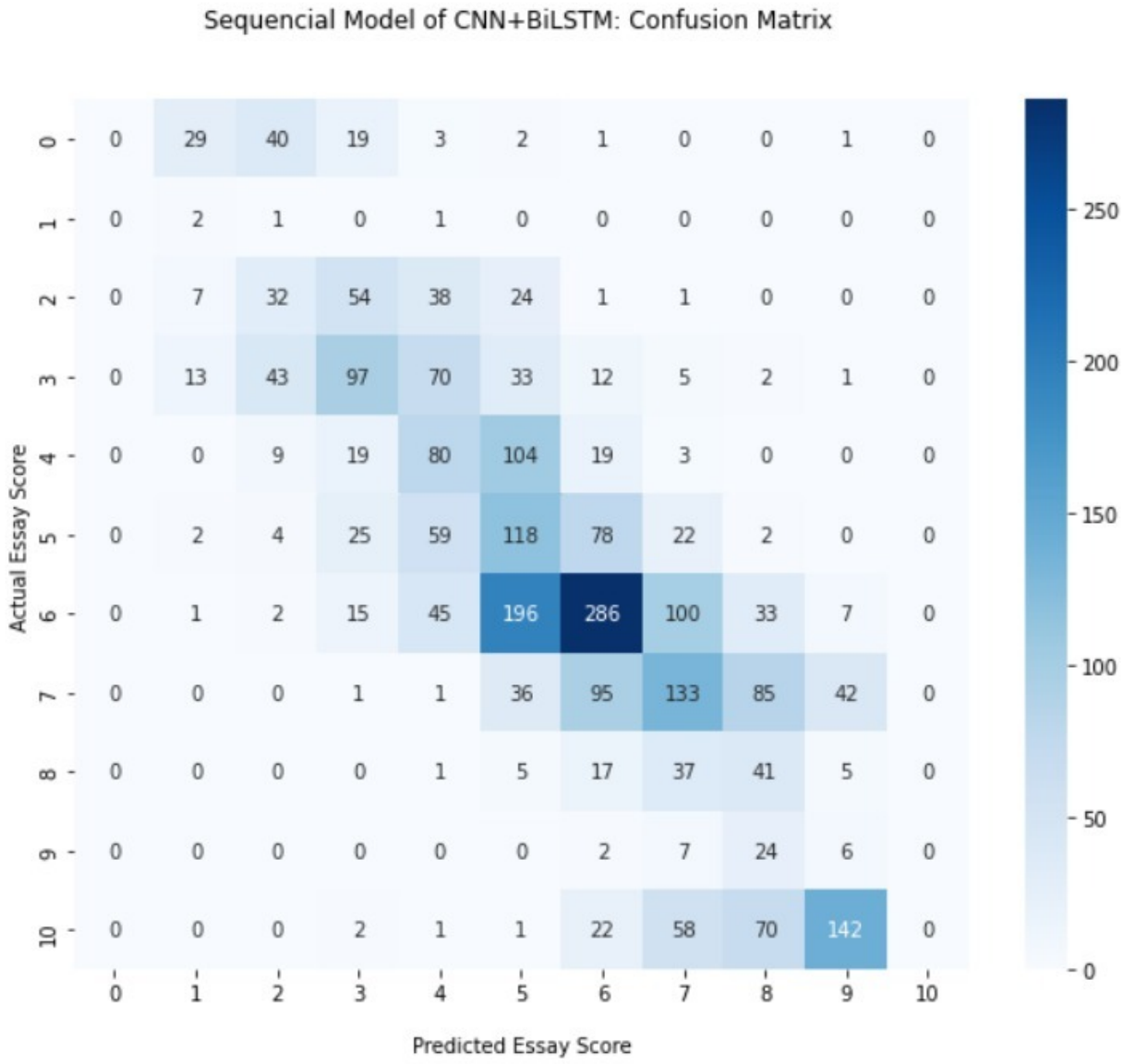

Figure 18: CNN+LSTM+BiLSTM Confusion Matrix

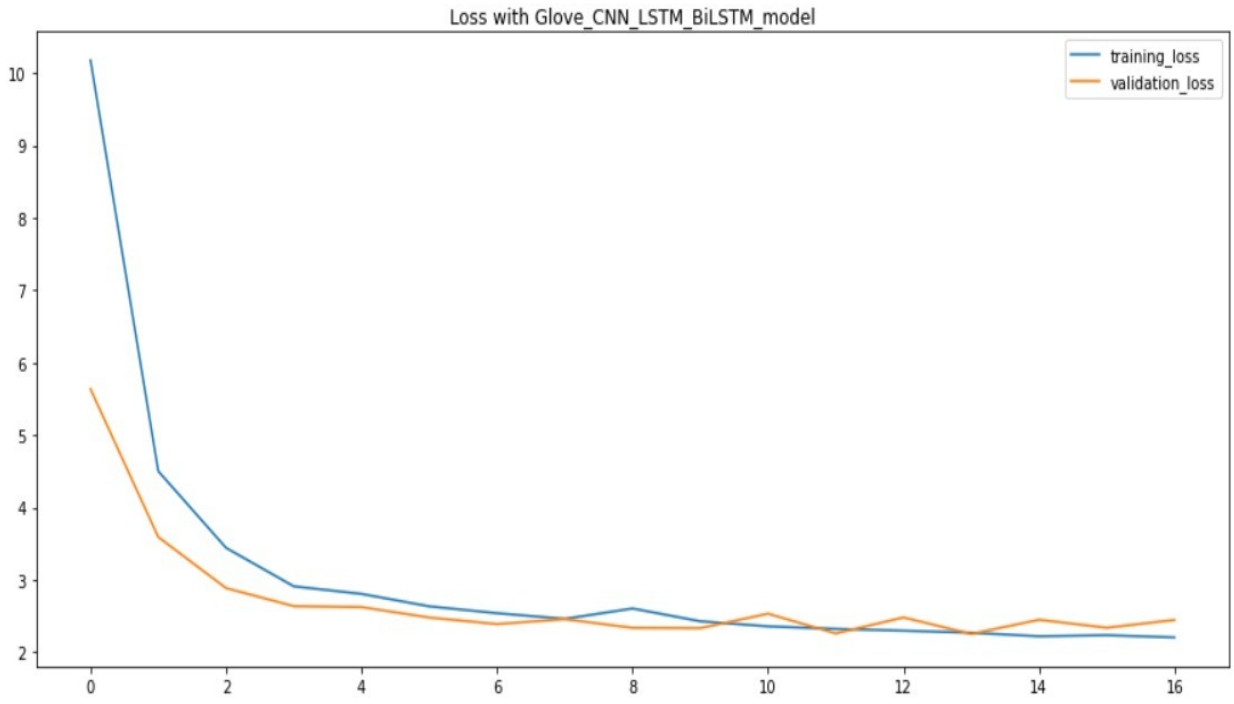

Figure 19: Learning Curve of CNN+LSTM+BiLSTM

	precision	recall	f1-score	support
0	0	0	0	95
1	0.043478	0.5	0.08	4
2	0.274194	0.216561	0.241993	157
3	0.408654	0.307971	0.35124	276
4	0.279693	0.311966	0.294949	234
5	0.204668	0.367742	0.262976	310
6	0.495274	0.382482	0.431631	685
7	0.346591	0.310433	0.327517	393
8	0.099656	0.273585	0.146096	106
9	0.008811	0.051282	0.015038	39
10	0	0	0	296
accuracy	0.278613	0.278613	0.278613	0.278613
macro avg	0.196456	0.247456	0.195585	2595
weighted avg	0.29722	0.278613	0.279865	2595

Table 10: Classification report of CNN+LSTM+BiLSTM

For the neural networks we have presented the confusion matrix in figure (18), learning curve in figure (19) and the classification report of the model in table (10). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has not given 0 to any of the essays instead the predicted score ranges [1-4]. It can be observed from the matrix that most of the actual score 6 is predicted 5. However, the highest predicted score is 9 which the given to score 10 essays. The weighted precision of the model is 0.29722 while the recall is 0.278613.

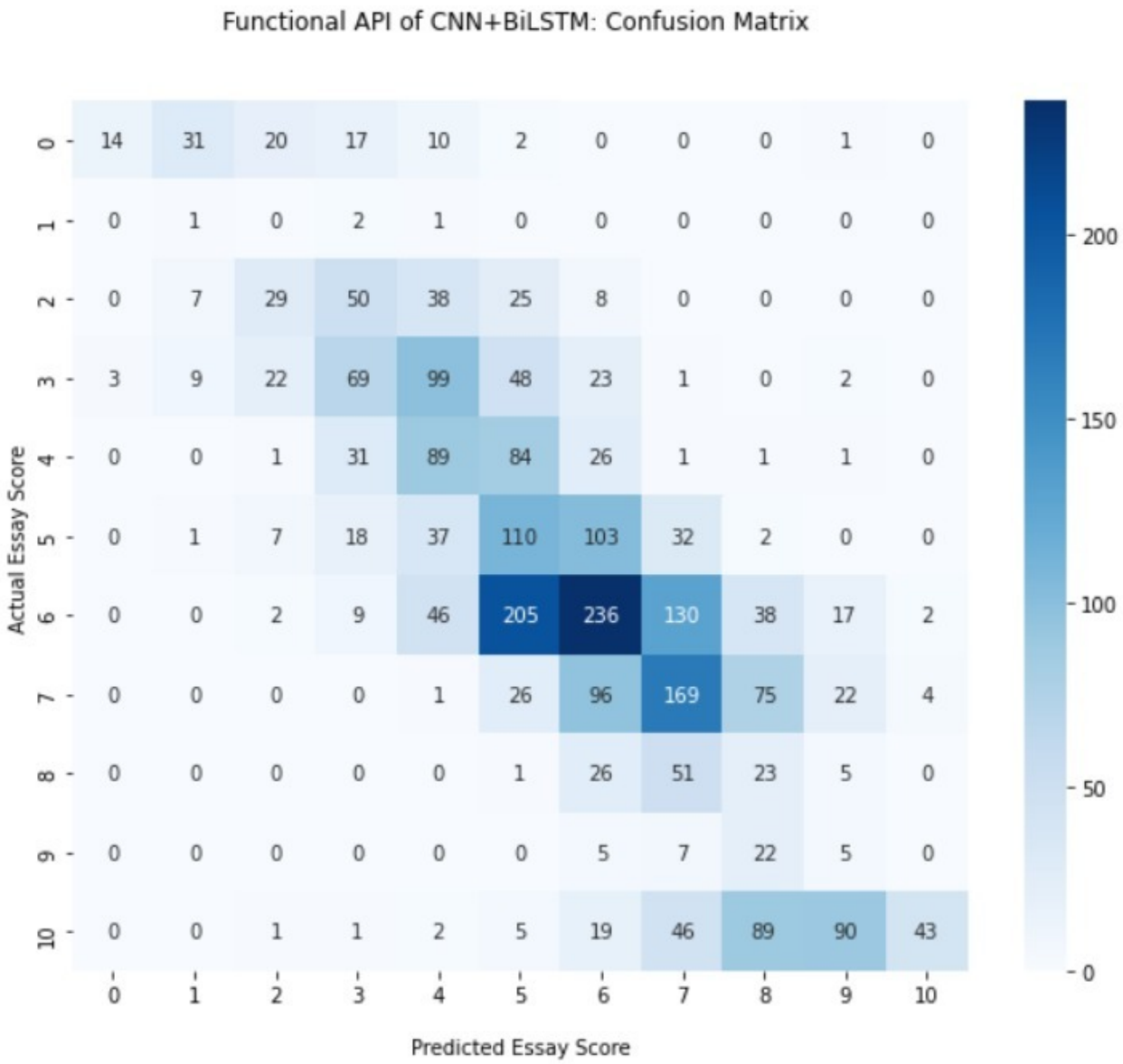

Figure 20: Functional API CNN+LSTM Confusion Matrix

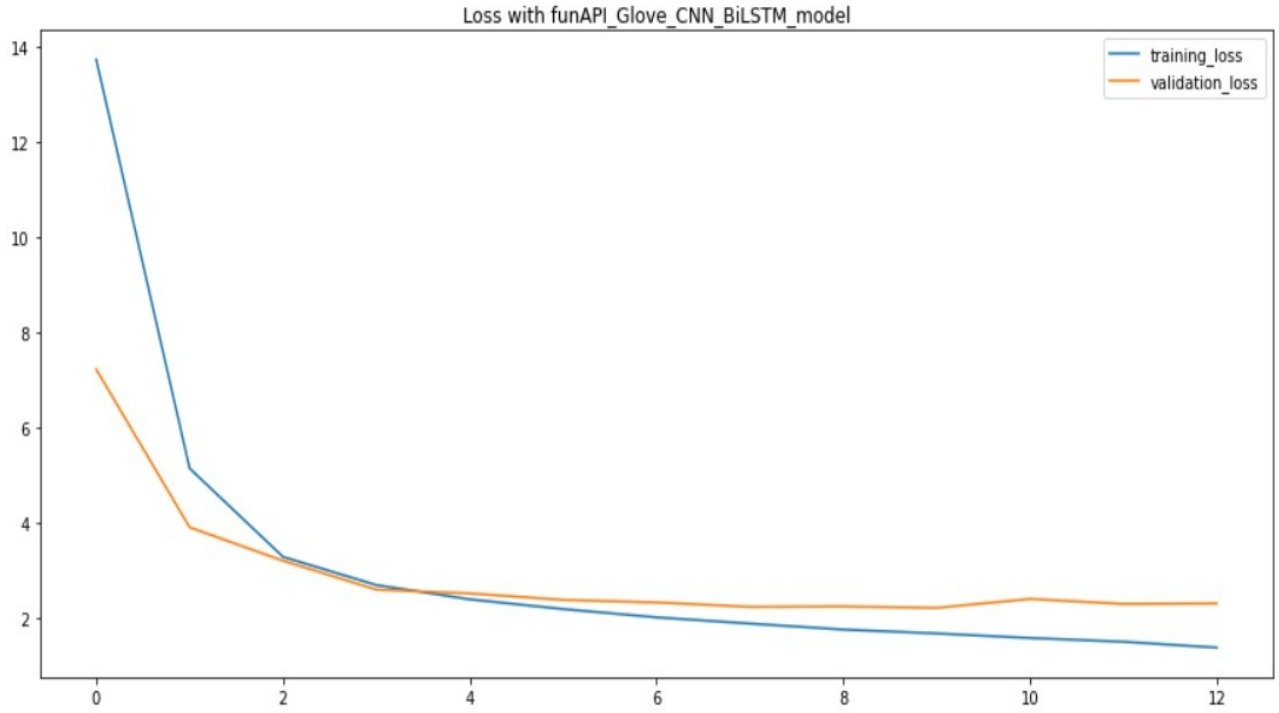

Figure 21: Learning Curve of functional API CNN+BiLSTM

	precision	recall	f1-score	support
0	0.769231	0.210526	0.330579	95
1	0.016129	0.25	0.030303	4
2	0.241758	0.140127	0.177419	157
3	0.410811	0.275362	0.329718	276
4	0.278146	0.358974	0.313433	234
5	0.235165	0.345161	0.279739	310
6	0.51	0.372263	0.43038	685
7	0.346698	0.374046	0.359853	393
8	0.146953	0.386792	0.212987	106
9	0.027149	0.153846	0.046154	39
10	0.857143	0.141892	0.243478	296
11	0	0	0	0
accuracy	0.308671	0.308671	0.308671	0.308671
macro avg	0.319932	0.250749	0.229504	2595
weighted avg	0.430991	0.308671	0.324903	2595

Table 11: Classification report of functional API of CNN+BiLSTM

For the neural networks we have presented the confusion matrix in figure (20), learning curve in figure (21) and the classification report of the model in table (11). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has scored the below mid score essays in range of [0-4] whose actual score was 0, while above mean score the models gives [6-10] to the essays whose actual score was 10. The weighted precision of the model is 0.430991 while the recall is 0.308671.

4.1.3 Trial 3: Transformers

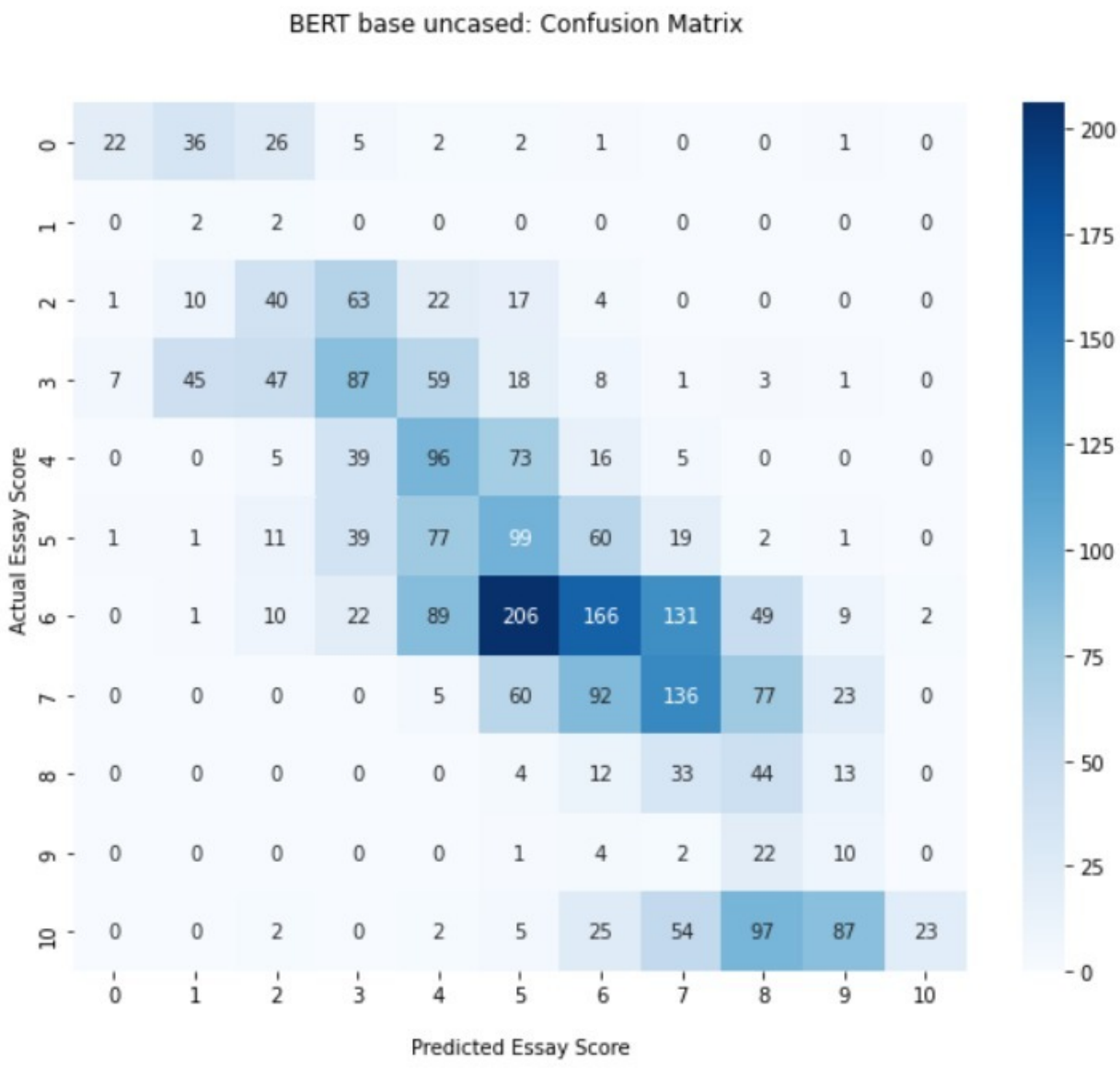

Figure 22: BERT uncased Confusion Matrix

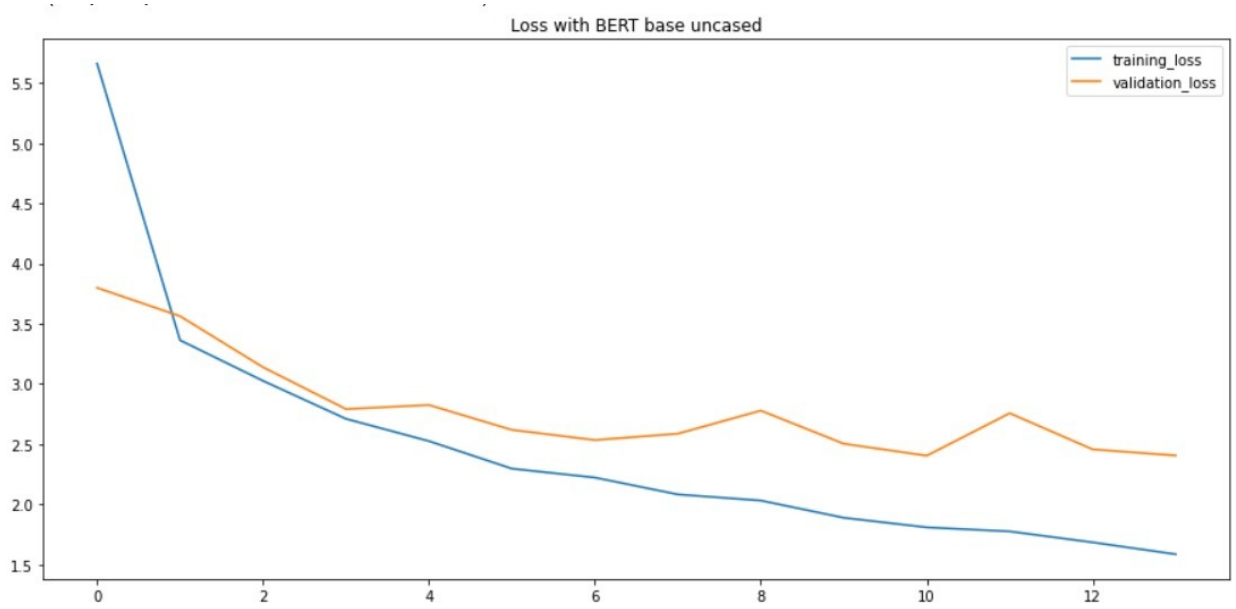

Figure 23: Learning curve of BERT uncased

	precision	recall	f1-score	support
0	0.709677	0.231579	0.349206	95
1	0.021053	0.5	0.040404	4
2	0.27972	0.254777	0.266667	157
3	0.341176	0.315217	0.327684	276
4	0.272727	0.410256	0.327645	234
5	0.204124	0.319355	0.249057	310
6	0.427835	0.242336	0.309413	685
7	0.356955	0.346056	0.351421	393
8	0.14966	0.415094	0.22	106
9	0.068966	0.25641	0.108696	39
10	0.92	0.077703	0.143302	296
accuracy	0.279383	0.279383	0.279383	0.279383
macro avg	0.312658	0.280732	0.224458	2595
weighted avg	0.407285	0.279383	0.284992	2595

Table 12: Classification report of BERT uncased

For the transformer neural networks, we have presented the confusion matrix in figure (22), learning curve in figure (23) and the classification report of the model in table (12). The

learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has scored the below mid score essays in range of [0-3] whose actual score was 0, while above mid score the models gives [6-10] to the essays whose actual score was 10. The weighted precision of the model is 0.407285 while the recall is 0.279383.

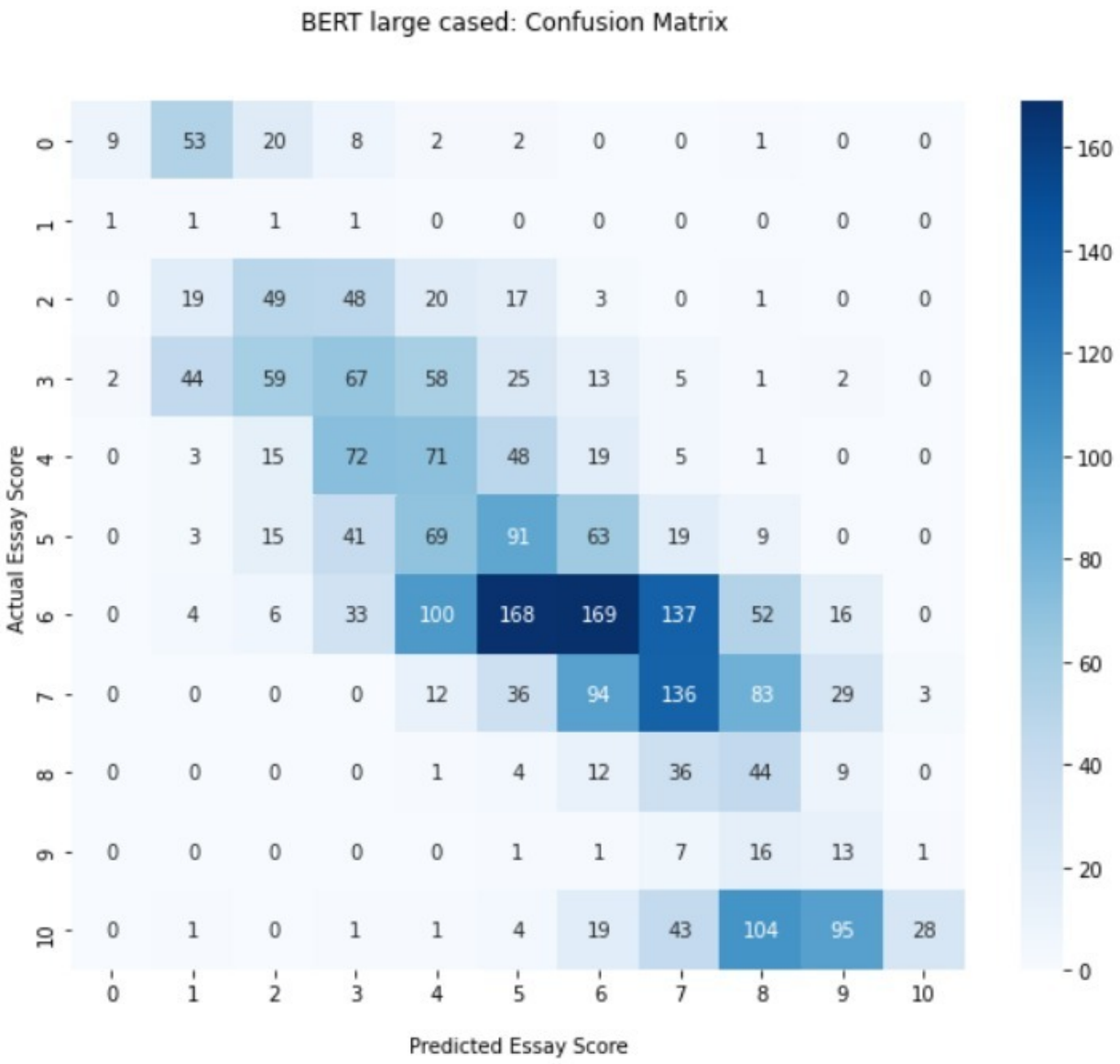

Figure 24: BERT large cased confusion matrix

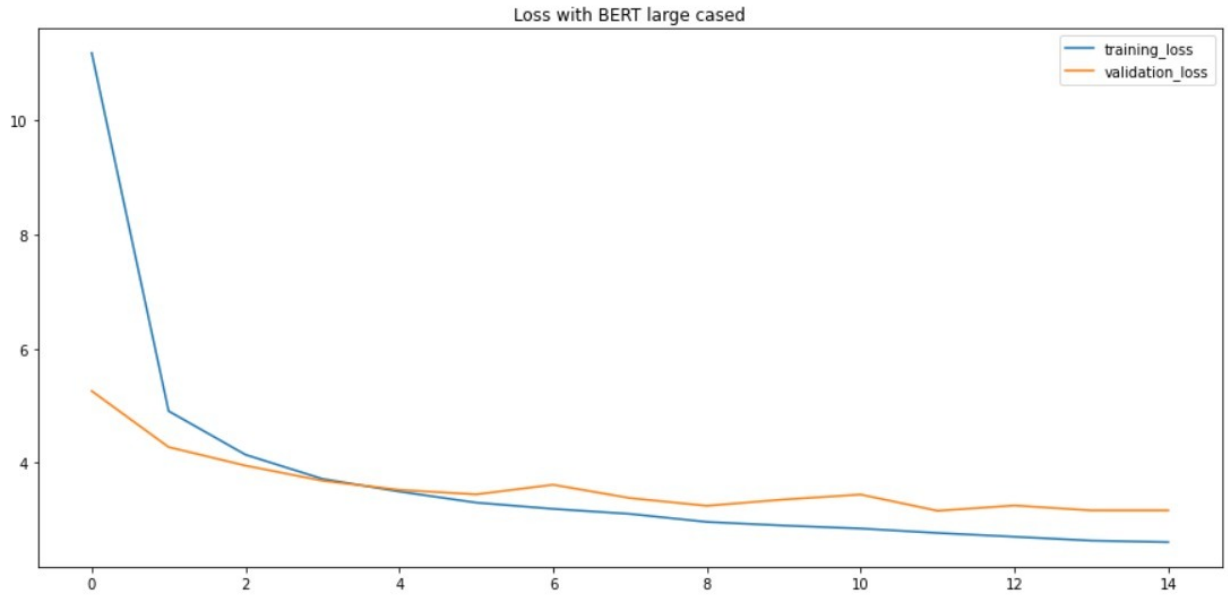

Figure 25: Learning curve of BERT large cased

	precision	recall	f1-score	support
0	0.75	0.094737	0.168224	95
1	0.007813	0.25	0.015152	4
2	0.29697	0.312102	0.304348	157
3	0.247232	0.242754	0.244973	276
4	0.212575	0.303419	0.25	234
5	0.229798	0.293548	0.25779	310
6	0.430025	0.246715	0.313544	685
7	0.350515	0.346056	0.348271	393
8	0.141026	0.415094	0.210526	106
9	0.079268	0.333333	0.128079	39
10	0.875	0.094595	0.170732	296
accuracy	0.261272	0.261272	0.261272	0.261272
macro avg	0.329111	0.266578	0.21924	2595
weighted avg	0.391708	0.261272	0.269498	2595

Table 13: classification report of BERT large cased

For the transformer neural networks, we have presented the confusion matrix in figure (24), learning curve in figure (25) and the classification report of the model in table (13). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has scored the below mid score essays in range of [1-5] whose actual score was 0, while above mid score the models gives [6-10] to the essays whose actual score was 10. The weighted precision of the model is 0.391708 while the recall is 0.261272.

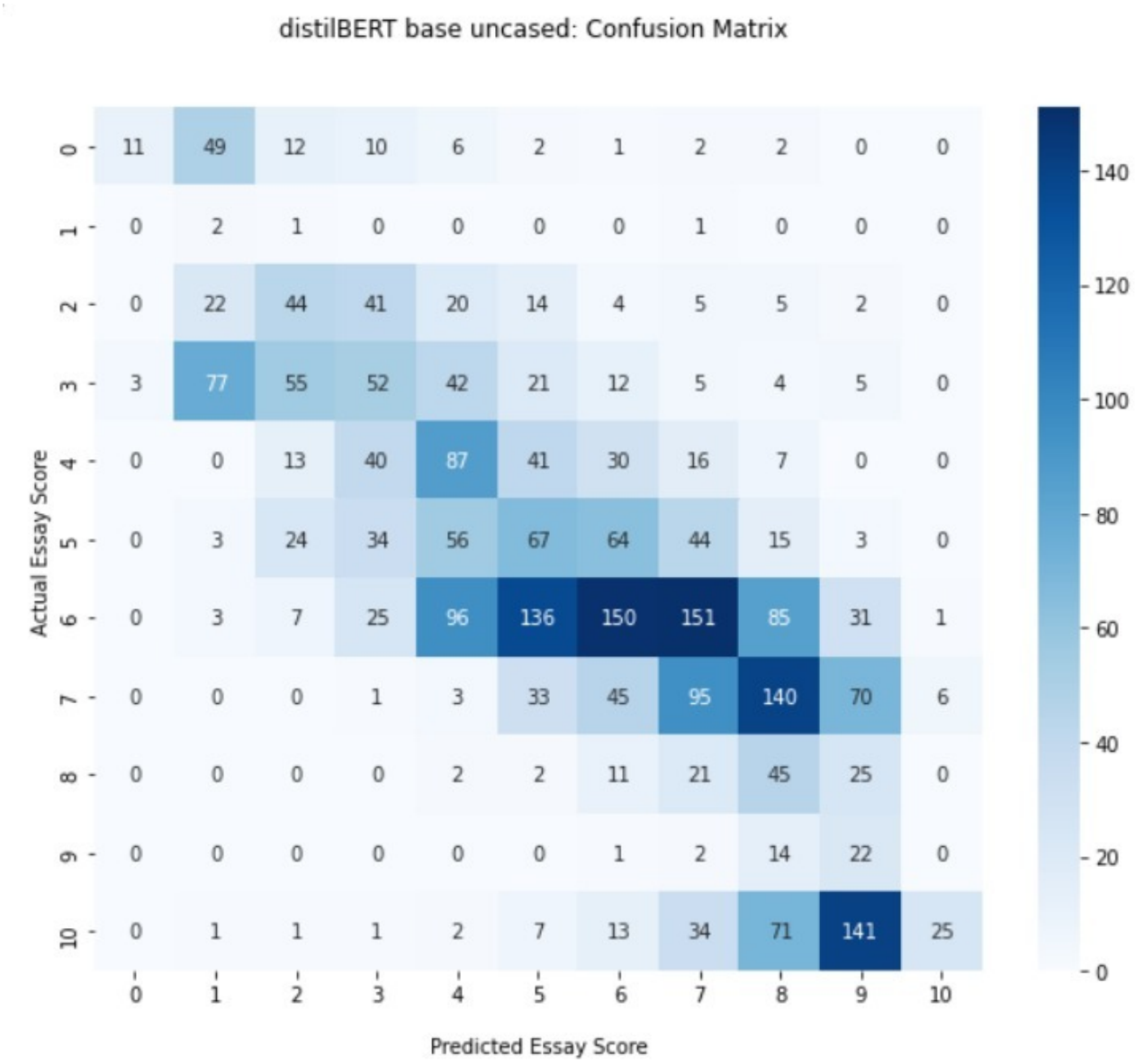

Figure 26: distillBERT base uncased confusion matrix

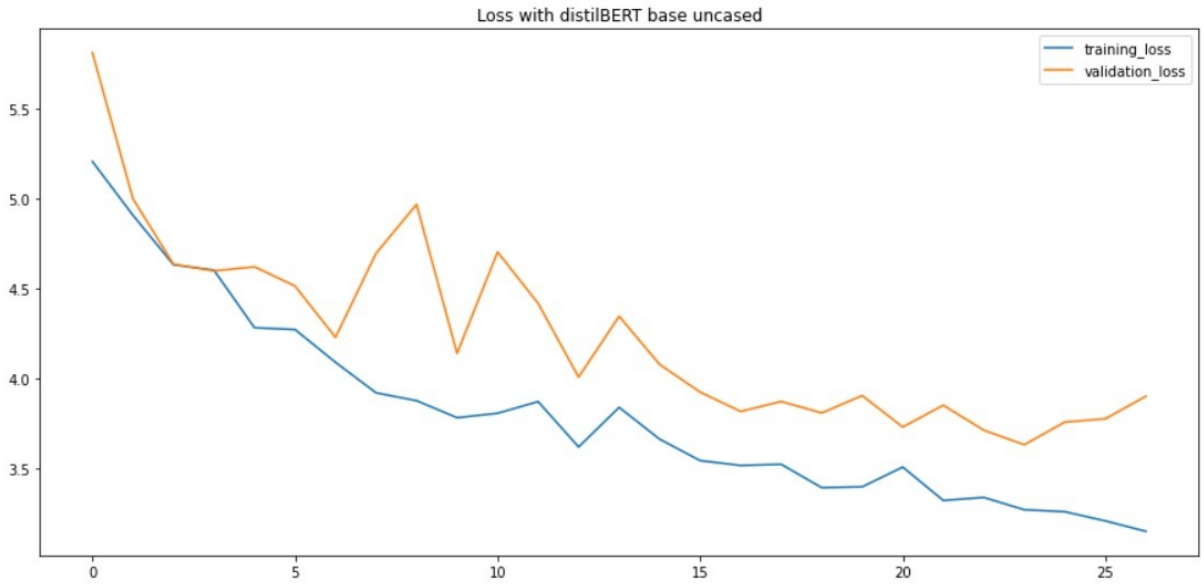

Figure 27: distillBERT base uncased learning curve

	precision	recall	f1-score	support
0	0.785714	0.115789	0.201835	95
1	0.012739	0.5	0.024845	4
2	0.280255	0.280255	0.280255	157
3	0.254902	0.188406	0.216667	276
4	0.27707	0.371795	0.317518	234
5	0.20743	0.216129	0.21169	310
6	0.453172	0.218978	0.295276	685
7	0.25266	0.24173	0.247074	393
8	0.115979	0.424528	0.182186	106
9	0.073579	0.564103	0.130178	39
10	0.78125	0.084459	0.152439	296
accuracy	0.231214	0.231214	0.231214	0.231214
macro avg	0.317705	0.29147	0.205451	2595
weighted avg	0.375459	0.231214	0.243496	2595

Table 14: classification report of distillBERT base uncased

For the transformer neural networks, we have presented the confusion matrix in figure (26), learning curve in figure (27) and the classification report of the model in table (14). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has scored the below mid score essays in range of [1-3] whose actual score was 0, while above mid score the models gives [6-10] to the essays whose actual score was 10. The weighted precision of the model is 0.375459 while the recall is 0.261272.

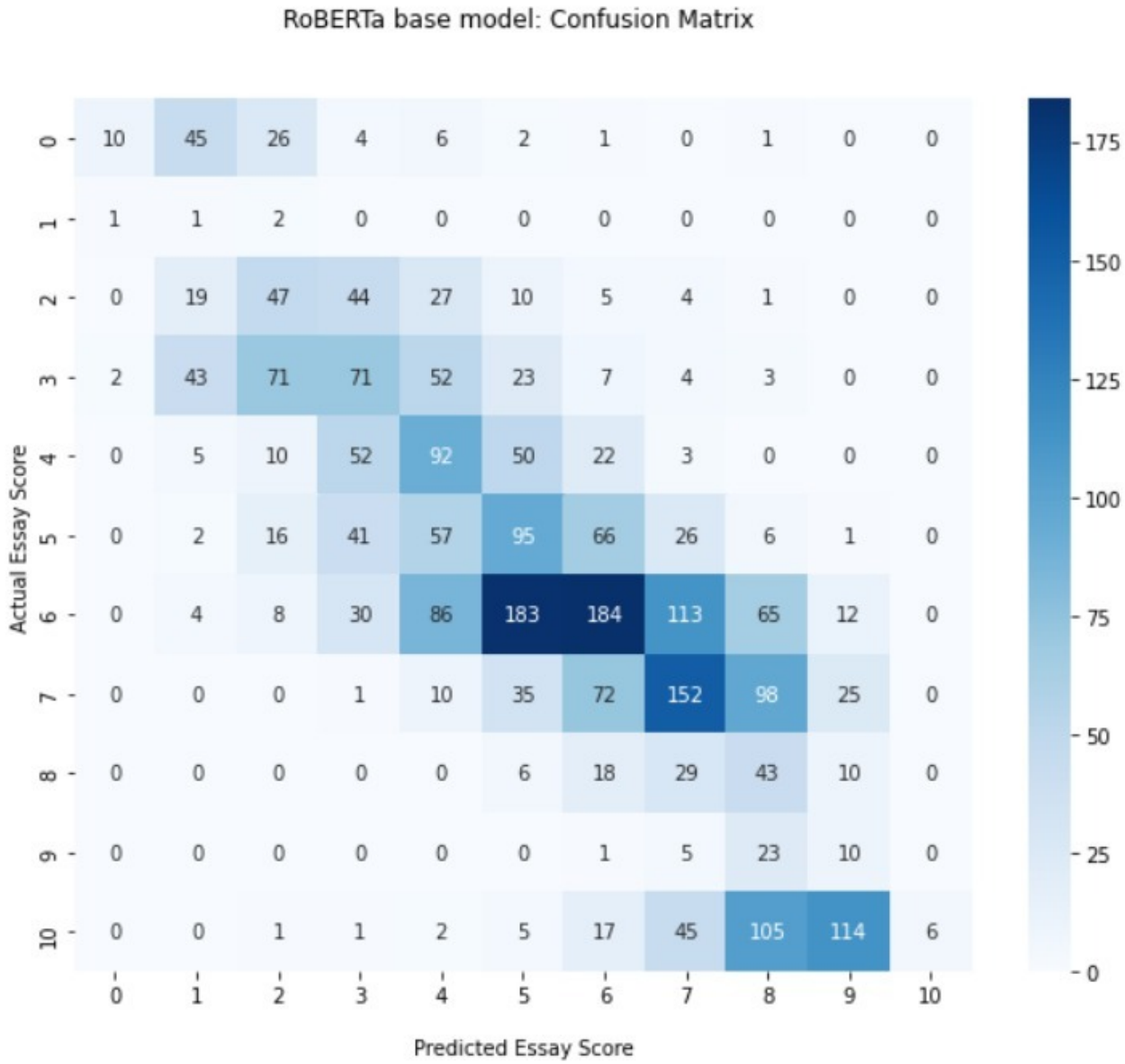

Figure 28: RoBERTa base confusion matrix

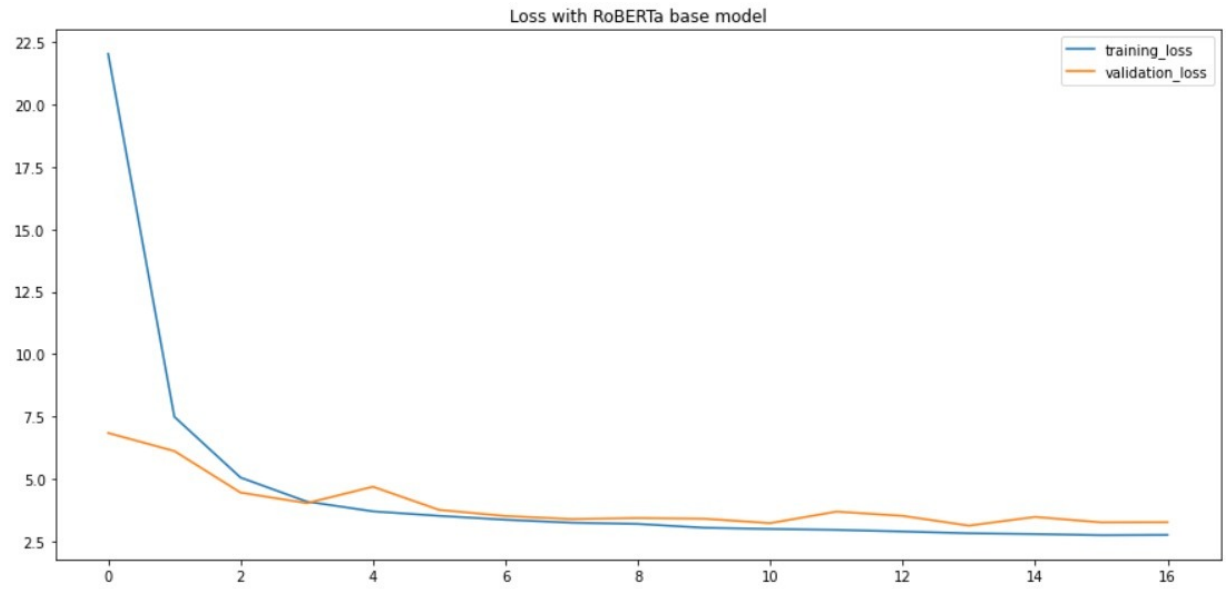

Figure 29: RoBERTa base learning curve

	precision	recall	f1-score	support
0	0.769231	0.105263	0.185185	95
1	0.008403	0.25	0.01626	4
2	0.259669	0.299363	0.278107	157
3	0.290984	0.257246	0.273077	276
4	0.277108	0.393162	0.325088	234
5	0.232274	0.306452	0.264256	310
6	0.468193	0.268613	0.341373	685
7	0.39895	0.386768	0.392765	393
8	0.124638	0.40566	0.190687	106
9	0.05814	0.25641	0.094787	39
10	1	0.02027	0.039735	296
accuracy	0.273988	0.273988	0.273988	0.273988
macro avg	0.353417	0.26811	0.218302	2595
weighted avg	0.431606	0.273988	0.276897	2595

Table 15: RoBERTa base classification report

For the transformer neural networks, we have presented the confusion matrix in figure (28), learning curve in figure (29) and the classification report of the model in table (15). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has scored the below mid score essays in range of [1-2] mostly whose actual score was 0, while above mid score the models gives [6-10] to the essays whose actual score was 10. The weighted precision of the model is 0.431606 while the recall is 0.273988.

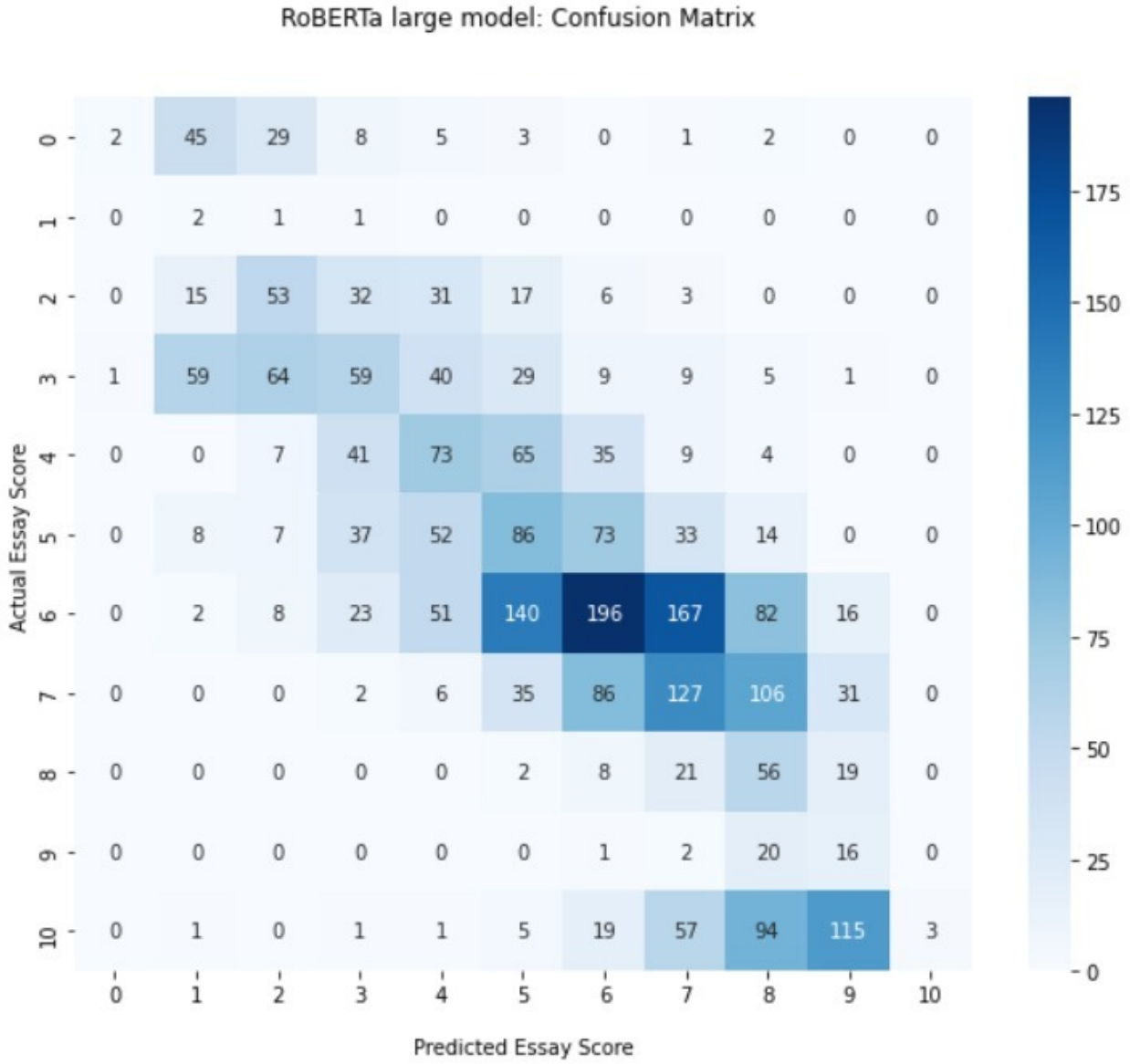

Figure 30: RoBERTa large confusion matrix

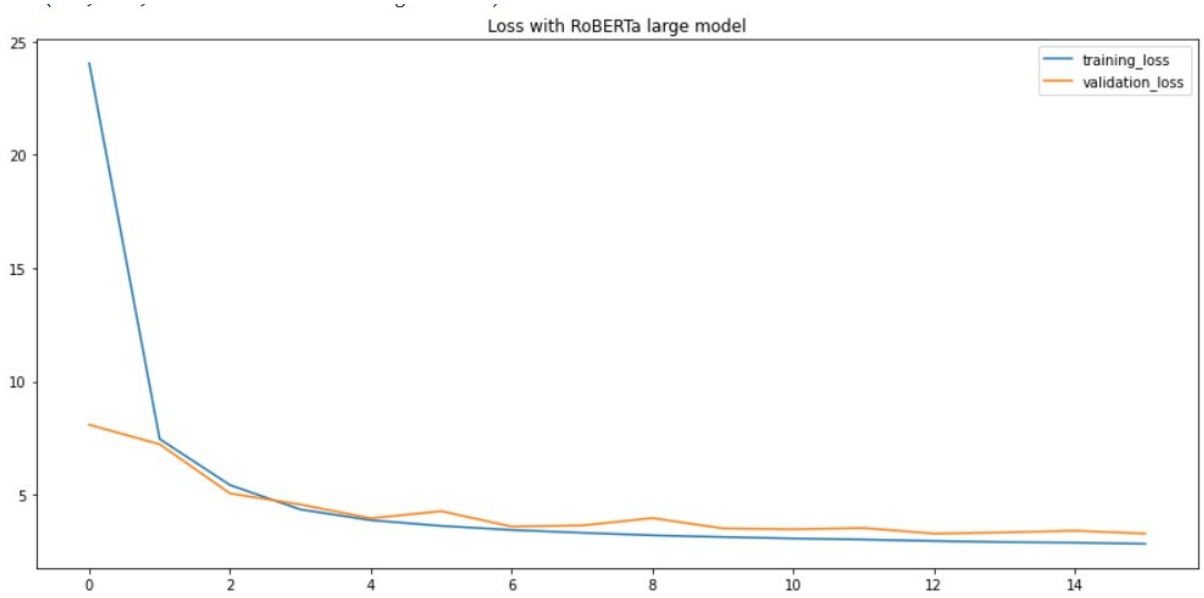

Figure 31: RoBERTa large learning curve

	precision	recall	f1-score	support
0	0.666667	0.021053	0.040816	95
1	0.015152	0.5	0.029412	4
2	0.313609	0.33758	0.325153	157
3	0.289216	0.213768	0.245833	276
4	0.281853	0.311966	0.296146	234
5	0.225131	0.277419	0.248555	310
6	0.452656	0.286131	0.350626	685
7	0.296037	0.323155	0.309002	393
8	0.146214	0.528302	0.229039	106
9	0.080808	0.410256	0.135021	39
10	1	0.010135	0.020067	296
accuracy	0.259345	0.259345	0.259345	0.259345
macro avg	0.342486	0.292706	0.202697	2595
weighted avg	0.412046	0.259345	0.25678	2595

Table 16: RoBERTa large classification report

For the transformer neural networks, we have presented the confusion matrix in figure (30), learning curve in figure (31) and the classification report of the model in table (16). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has scored the below mid score essays in range of [1-2] mostly whose actual score was 0, while above mid score the models gives [7-10] to the essays whose actual score was 10. The weighted precision of the model is 0.412046 while the recall is 0.259345.

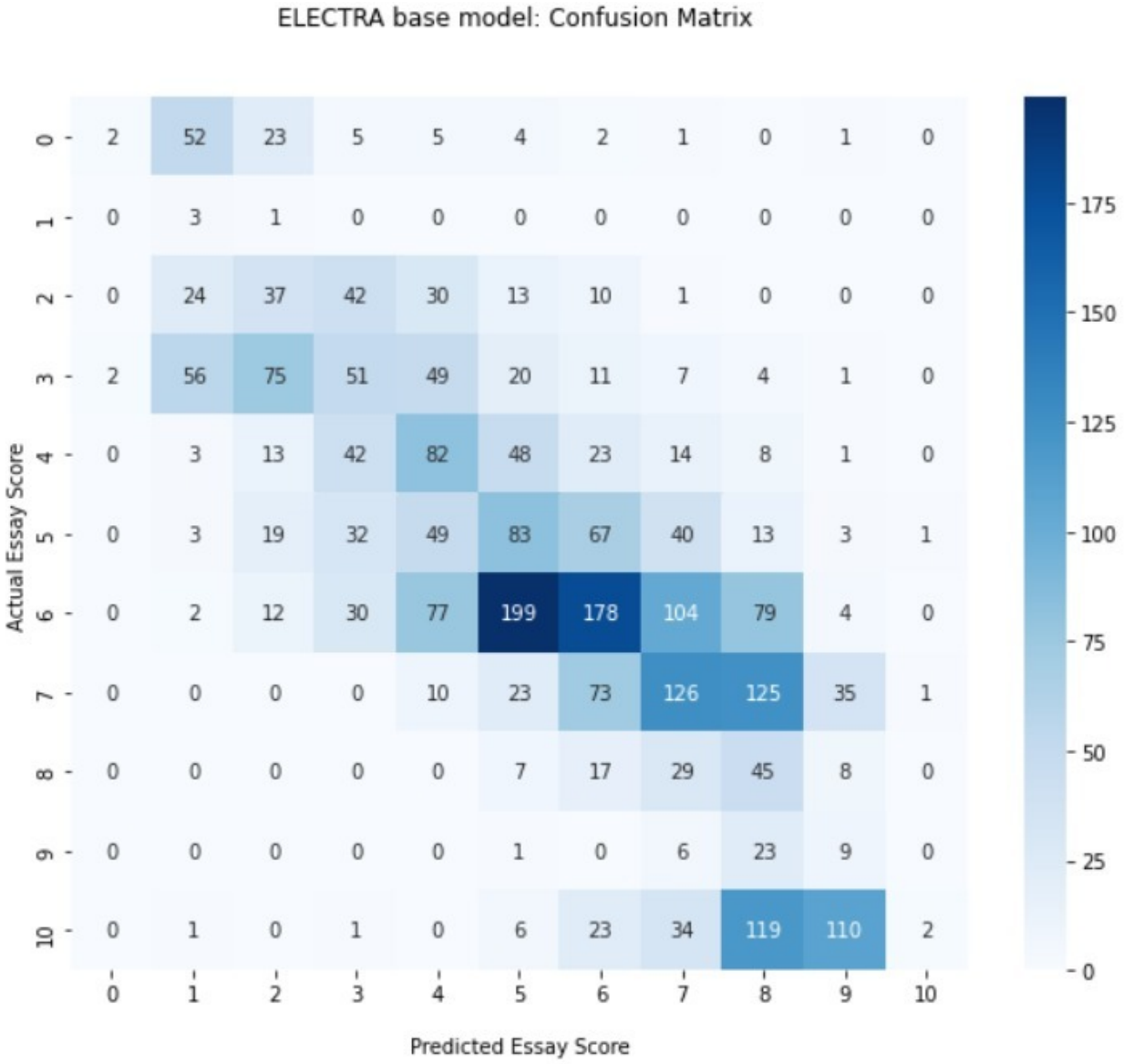

Figure 32: ELECTRA base confusion matrix

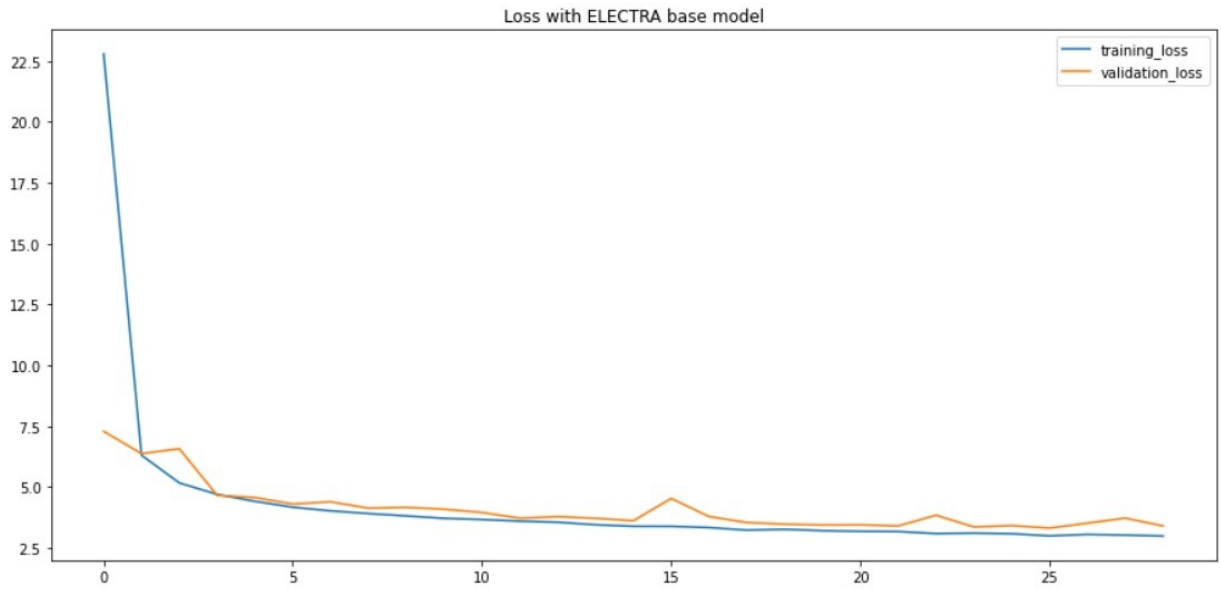

Figure 33: ELECTRA base learning curve

	precision	recall	f1-score	support
0	0.5	0.021053	0.040404	95
1	0.020833	0.75	0.040541	4
2	0.205556	0.235669	0.219585	157
3	0.251232	0.184783	0.212944	276
4	0.271523	0.350427	0.30597	234
5	0.205446	0.267742	0.232493	310
6	0.440594	0.259854	0.326905	685
7	0.348066	0.320611	0.333775	393
8	0.108173	0.424528	0.172414	106
9	0.052326	0.230769	0.085308	39
10	0.5	0.006757	0.013333	296
accuracy	0.23815	0.23815	0.23815	0.23815
macro avg	0.263977	0.277472	0.180334	2595
weighted avg	0.337774	0.23815	0.239526	2595

Table 17: ELECTRA base classification report

For the transformer neural networks, we have presented the confusion matrix in figure (32), learning curve in figure (33) and the classification report of the model in table (17). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has mapped the actual score in range of [2-3] to [1-6], while above mid score the models gives [6-9] to the essays whose actual score was 10. The weighted precision of the model is 0.337774 while the recall is 0.23815.

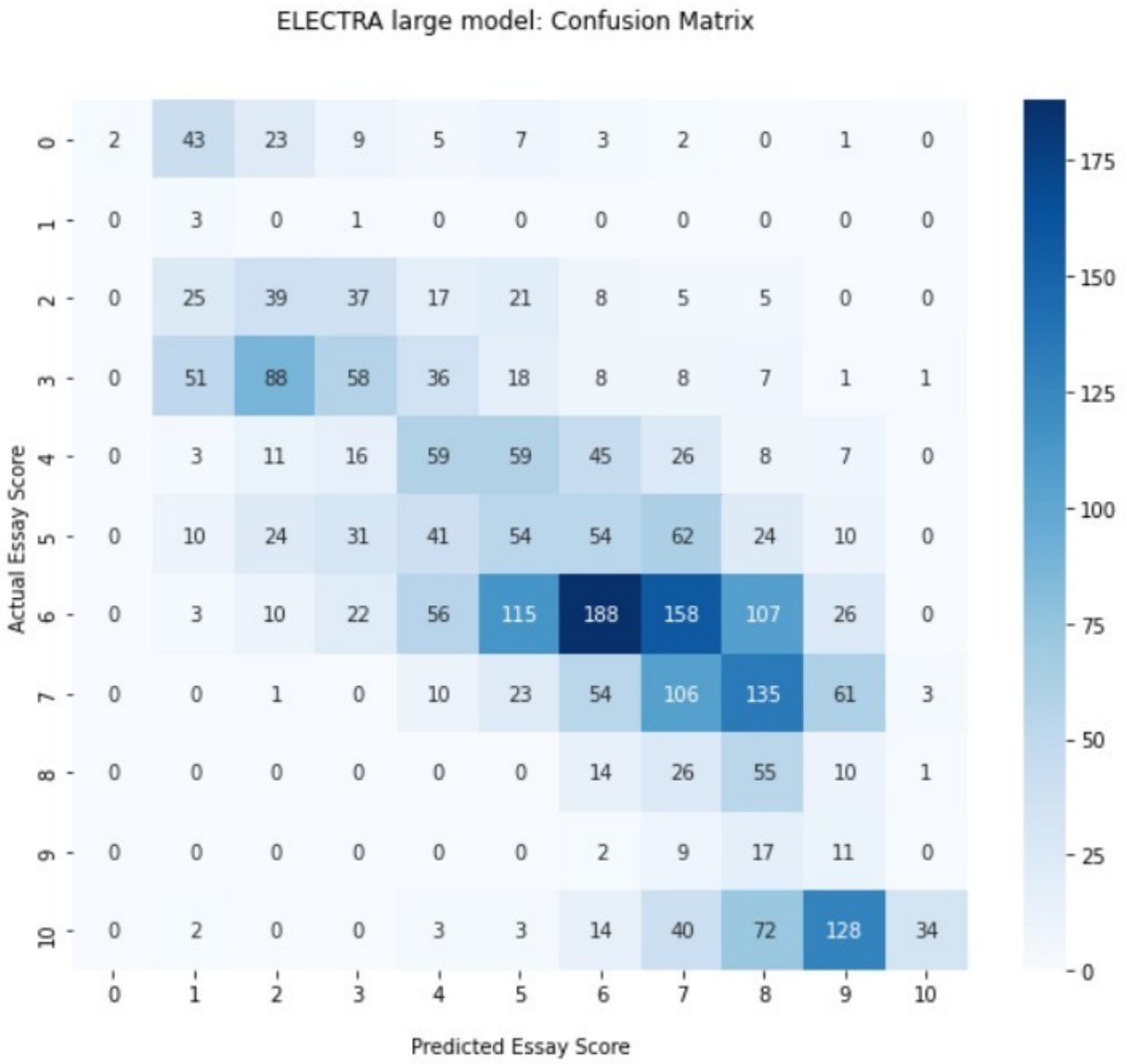

Figure 34: ELECTRA large confusion matrix

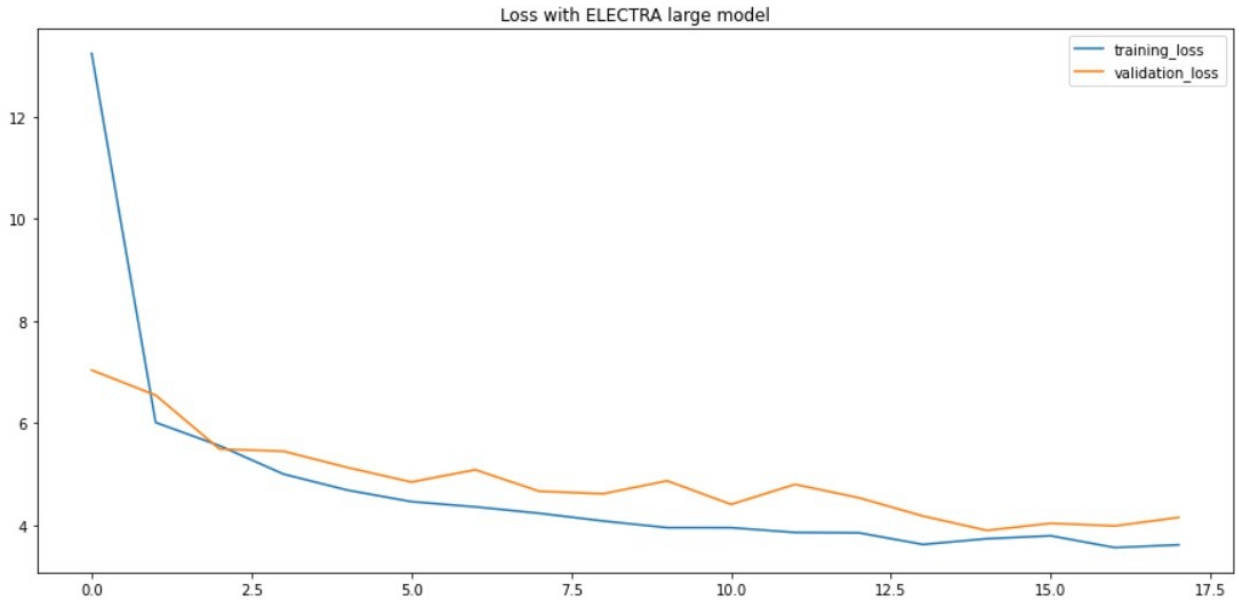

Figure 35:Learning curve of ELECTRA large

	precision	recall	f1-score	support
0	1	0.021053	0.041237	95
1	0.021429	0.75	0.041667	4
2	0.19898	0.248408	0.220963	157
3	0.333333	0.210145	0.257778	276
4	0.259912	0.252137	0.255965	234
5	0.18	0.174194	0.177049	310
6	0.482051	0.274453	0.349767	685
7	0.239819	0.26972	0.253892	393
8	0.127907	0.518868	0.205224	106
9	0.043137	0.282051	0.07483	39
10	0.871795	0.114865	0.202985	296
accuracy	0.234682	0.234682	0.234682	0.234682
macro avg	0.341669	0.283263	0.189214	2595
weighted avg	0.397954	0.234682	0.250031	2595

Table 18: ELECTRA large classification report

For the transformer neural networks, we have presented the confusion matrix in figure (34), learning curve in figure (35) and the classification report of the model in table (18). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has mapped the actual score in range of [2-3] to [2-5], while above mid score the models gives [6-10] to the essays whose actual score was 10. The weighted precision of the model is 0.397954 while the recall is 0.234682.

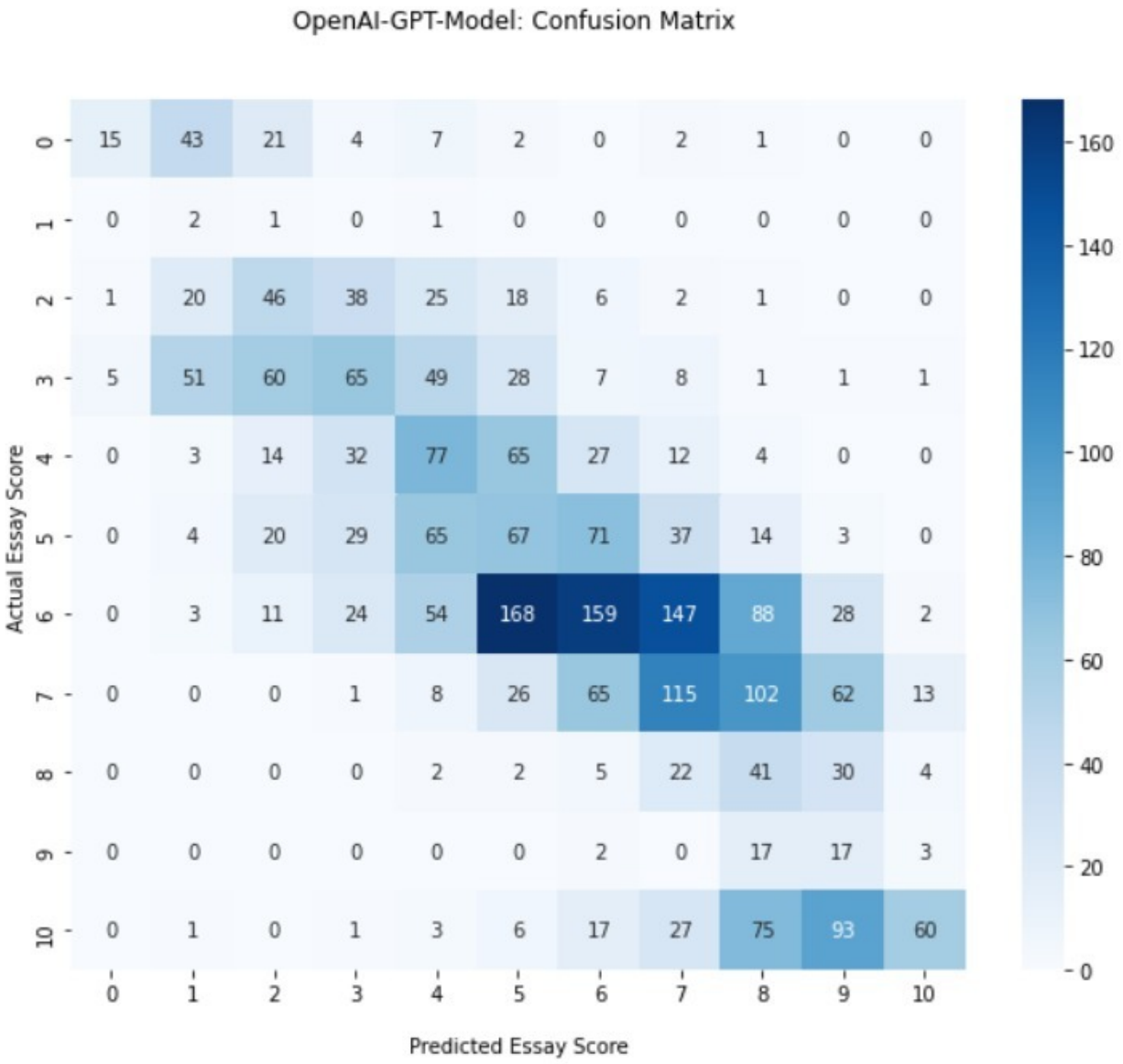

Figure 36: OpenAI GPT confusion matrix

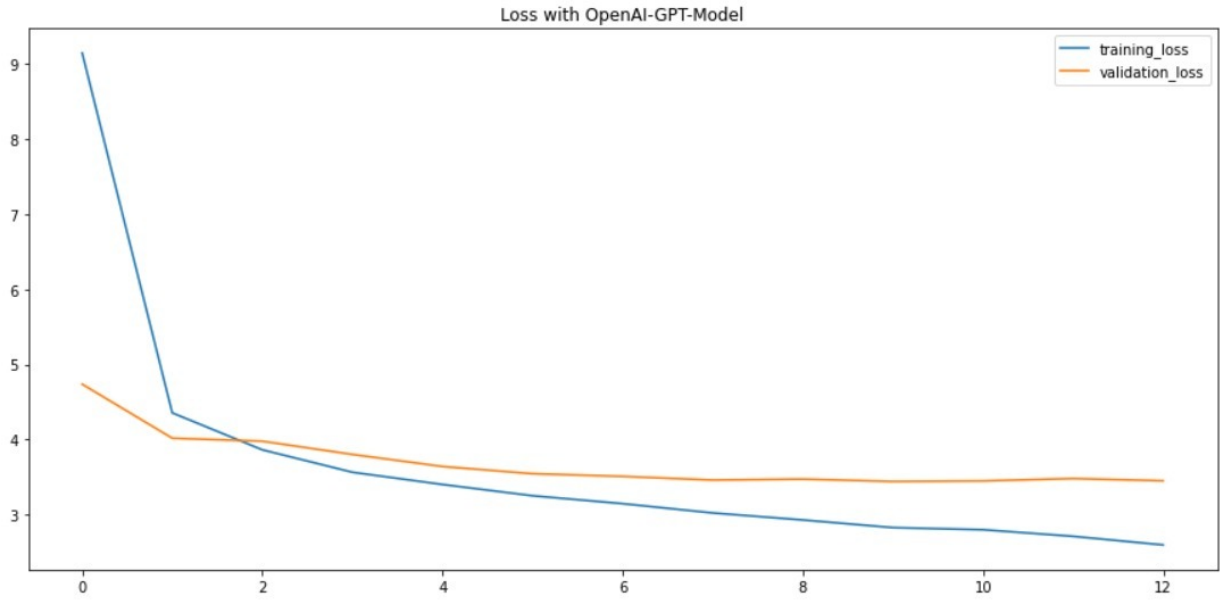

Figure 37: OpenAI GPT learning curve

	precision	recall	f1-score	support
0	0.714286	0.157895	0.258621	95
1	0.015748	0.5	0.030534	4
2	0.265896	0.292994	0.278788	157
3	0.335052	0.235507	0.276596	276
4	0.264605	0.32906	0.293333	234
5	0.175393	0.216129	0.193642	310
6	0.442897	0.232117	0.304598	685
7	0.30914	0.292621	0.300654	393
8	0.119186	0.386792	0.182222	106
9	0.07265	0.435897	0.124542	39
10	0.722892	0.202703	0.316623	296
accuracy	0.255877	0.255877	0.255877	0.255877
macro avg	0.286479	0.273476	0.213346	2595
weighted avg	0.374855	0.255877	0.276751	2595

Table 19: OpenAI GPT classification report

For the transformer neural networks, we have presented the confusion matrix in figure (36), learning curve in figure (37) and the classification report of the model in table (19). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.The confusion matrix shows that the model has mapped the actual score in range of [2-3] to [2-5] and [4-6] to [2-8], while above mid score the models gives [8-10] to the essays whose actual score was 10. The weighted precision of the model is 0.397954 while the recall is 0.234682.

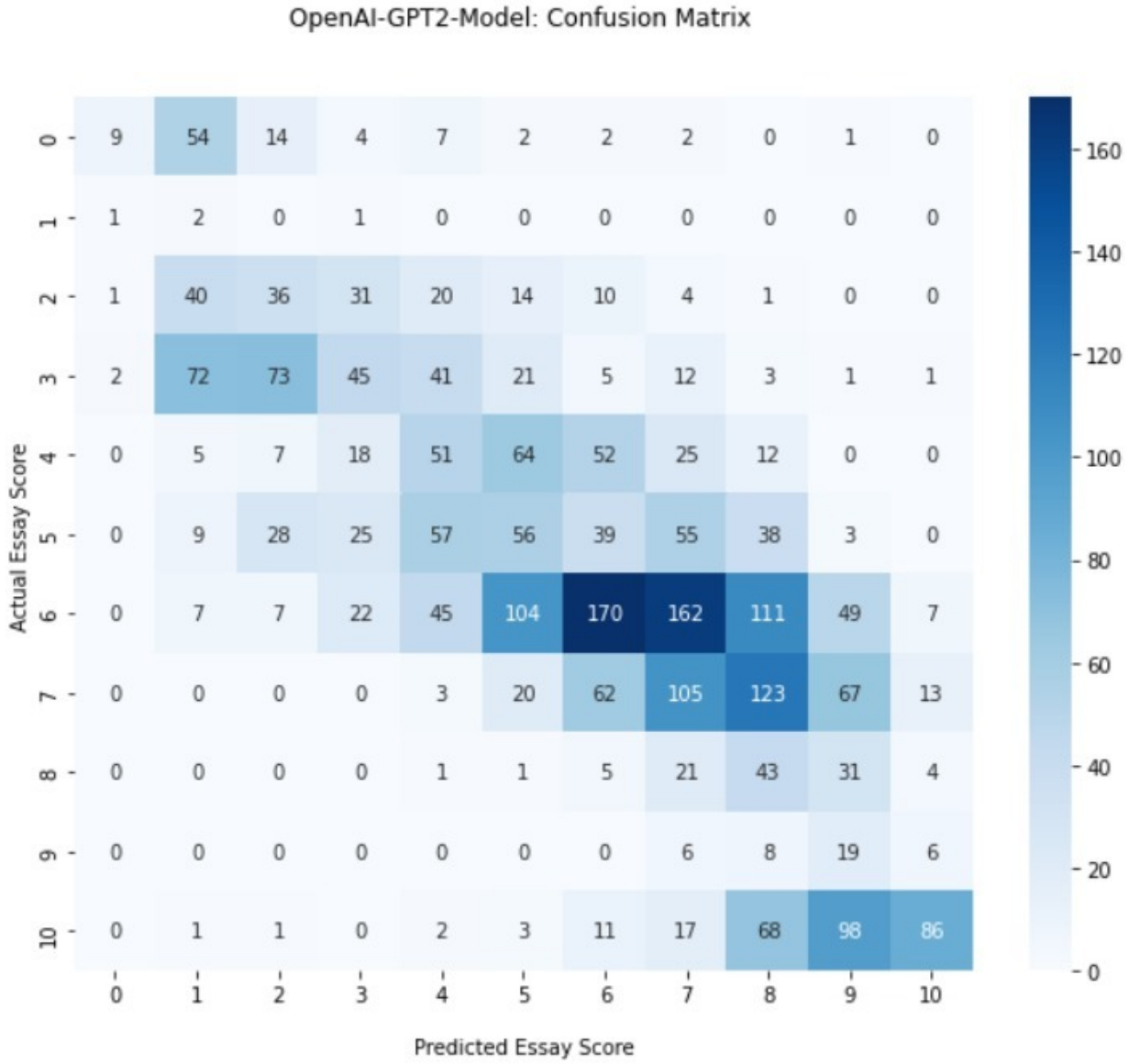

Figure 38: OpenAI GPT2 confusion matrix

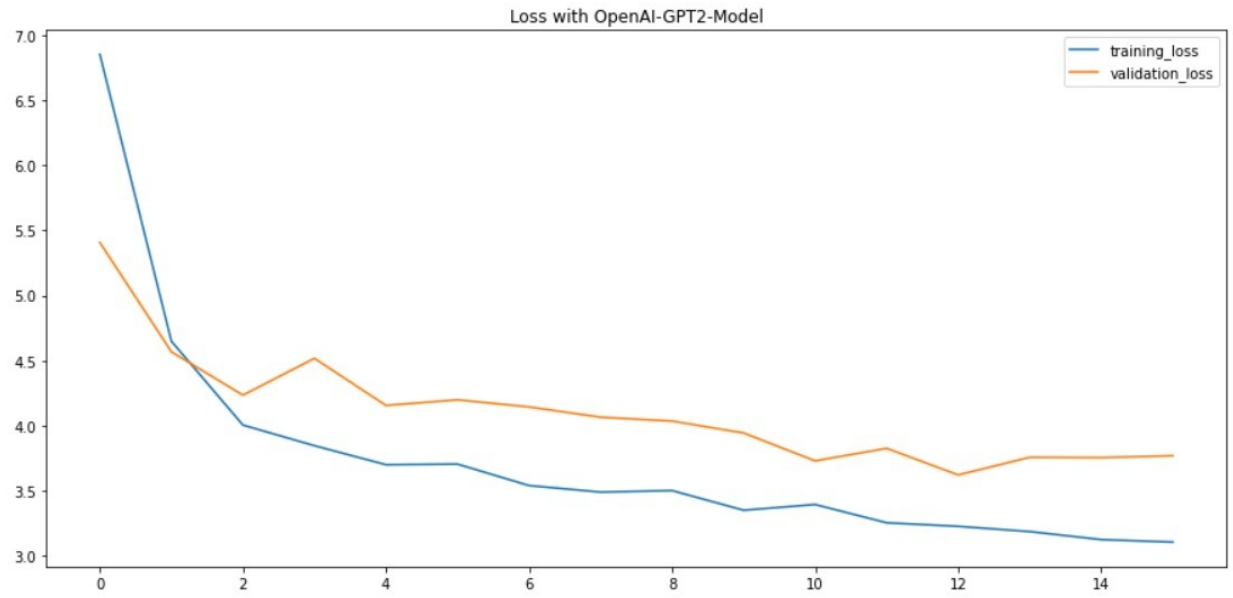

Figure 39: OpenAI GPT2 learning curve

	precision	recall	f1-score	support
0	0.692308	0.094737	0.166667	95
1	0.010526	0.5	0.020619	4
2	0.216867	0.229299	0.22291	157
3	0.308219	0.163043	0.21327	276
4	0.22467	0.217949	0.221258	234
5	0.196491	0.180645	0.188235	310
6	0.477528	0.248175	0.326609	685
7	0.256724	0.267176	0.261845	393
8	0.105651	0.40566	0.167641	106
9	0.070632	0.487179	0.123377	39
10	0.735043	0.290541	0.416465	296
accuracy	0.239692	0.239692	0.239692	0.239692
macro avg	0.274555	0.257034	0.194075	2595
weighted avg	0.369148	0.239692	0.266817	2595

Table 20: OpenAI GPT2 classification report

For the neural networks we have presented the confusion matrix in figure (38), learning curve in figure (39) and the classification report of the model in table (20). The learning curve is generated by using call backs and also the exact epoch count is not noted because of the early stopping.

The confusion matrix shows that the model has mapped the actual score in range of [2-3] to [3-5], while above mid score the models gives [8-10] to the essays whose actual score was 10. The weighted precision of the model is 0.369148 while the recall is 0.239692.

4.2 Summary of Test Results

This section covers the discussion and analysis of the results of the project. It includes the mean square error and root mean square error of the models which was trained on the dataset. It also presents the kappa score and precision, recall of the models.

The reason for not using the accuracy as a measure of model evaluation is that an overwhelming number of the majority of the classes will overwhelm the number of examples from the minority classes. Even the unskillful model can achieve an accuracy of above 90 percent depending on the class imbalance it has.

The precision shows the rate of the positive classes that belong to the positive class; the higher the precision the better. In comparison, recall relates to the number of positive class predictions made out of all the positive examples in the dataset. The higher the value of recall the good classification will be.

The discussion is divided into 3 sections and all sections shows the results of the same measure validation. The mean square error (MSE) shows the how close the predictions are to the actual values from the dataset. The lower the MSE the better the model is as the predictions will be closer to the actual labels. MSE is used as a model evaluation measure and lowers values to show a better fit. In addition to MSE root mean square error or RMSE is also used and RMSE is always smaller than MSE (MSE>RMSE). Both of the measures are sensitive to the outliers and penalize the larger error. The values are in the range of [0-∞].

Cohen's kappa statistic measures the degree of interrater reliability as shown in Table 21. If the two raters can use the same criterion for the same assessment on the same class, the agreement will be very high. Kappa score varies from a range of [0,1], and the score chart is as follows:

Kappa Value	**Level of Agreement**
0	No agreement
0.1-0.20	Minor agreement
0.21-0.40	Weak agreement
0.41-0.60	Reasonable agreement
0.61-0.80	Significant agreement
0.81-0.99	Strong agreement
1	Perfect agreement

Table 21: Cohen's Kappa score chart

4.2.1 Trial 1 – Classical Models

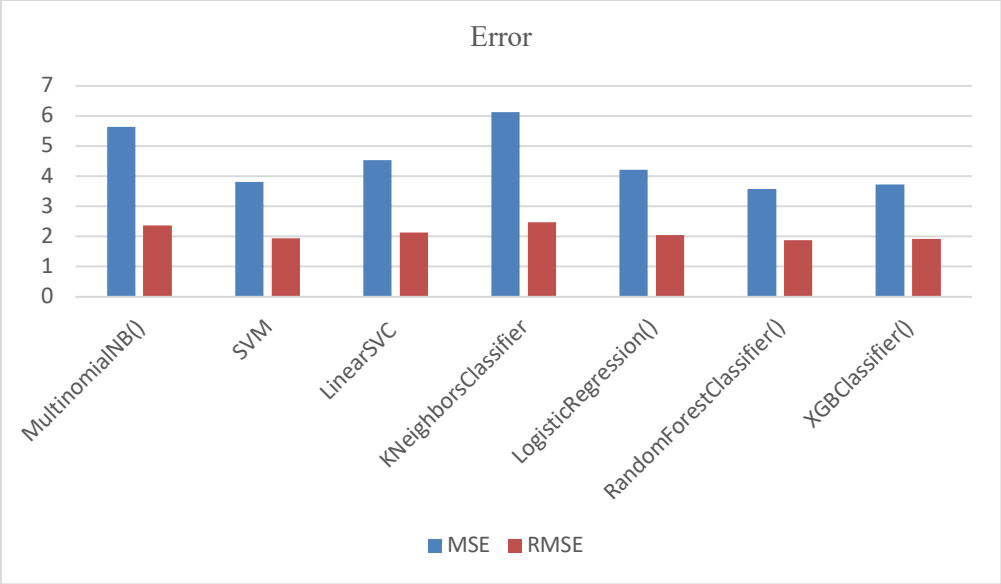

Figure 40: Traditional Classifiers Error count

Figure (40) illustrates that the MSE and RMSE values are high in KNN classifier i.e. 6.1 and 2.4 while Multinomial Naïve Bayes follows second highest error count which is 5.6 and 2.3 respectively. From the plot we observe that SVM has the lowest count out of all with 3.8 and 1.9 MSE and RMSE respectively.

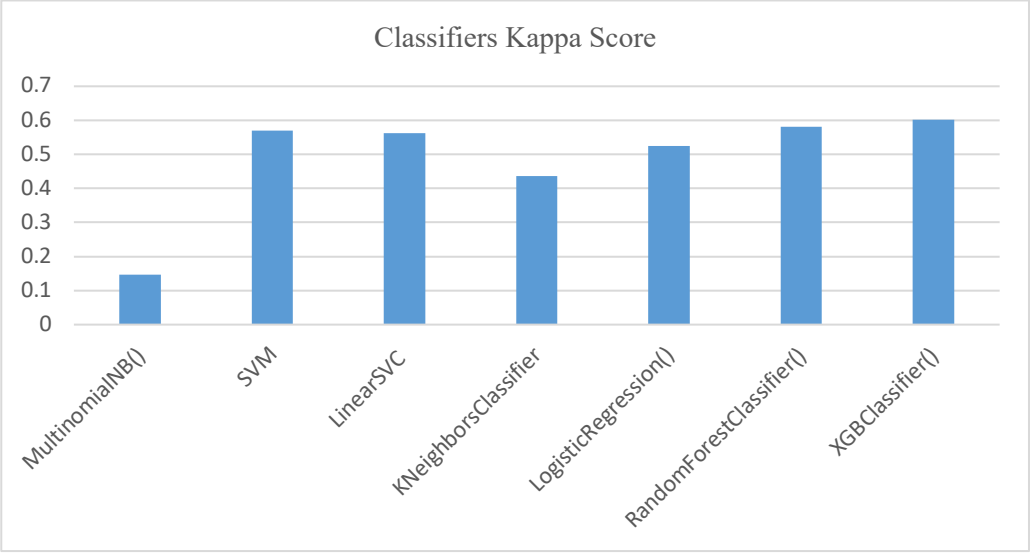

Figure 41: Kappa Score in Traditional Classifiers

Figure (41) illustrates the kappa score of different models. On more than 12000 essays dataset the models XGB classifier shows the .60 score being the highest while the lowest

agreement score was 0.14 of multinomial naïve bayes. The model which the scores of the XGB is SVM and Random Forest Classifier having score of 0.57 and 0.58 respectively while linear SVC has also in same tier with 0.56 score.

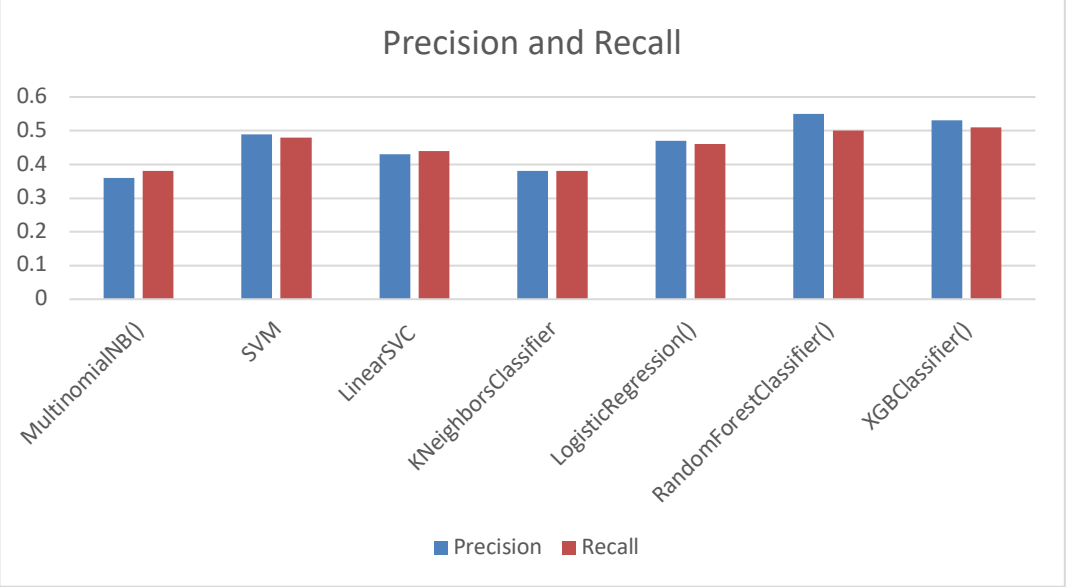

Figure 42: Traditional Classifiers Precision and Recall

Figure (42) illustrates that the random forest classifier has the highest precision, i.e. 0.55 and recall of 0.50. the model labelled 50% of the values correctly labelled and predicted. The XGB follows the random forest classifier with precision and recall of 0.53 and 0.51, respectively.

4.2.2 Trial 2 – Neural Networks

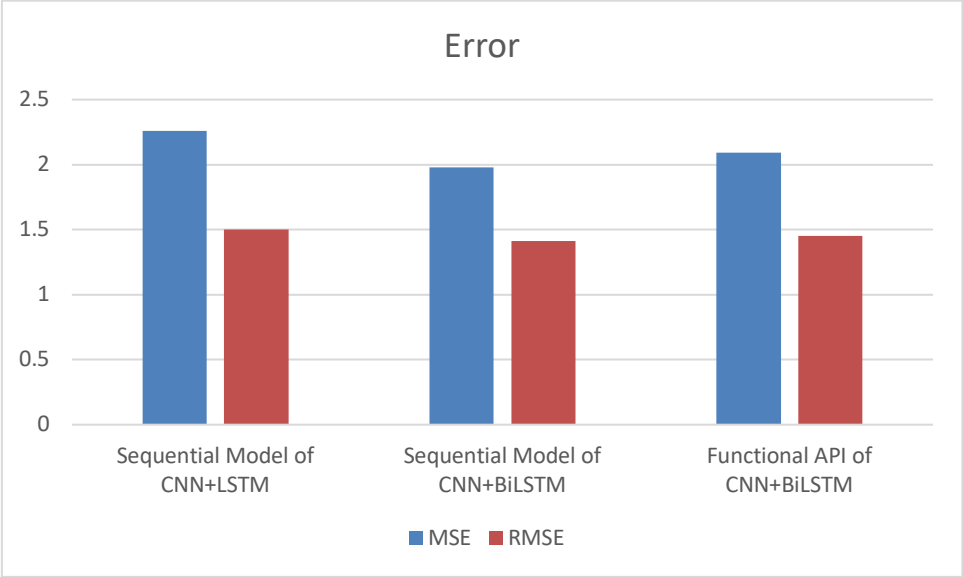

Figure 43: Neural Networks Error count

Figure (43) shows that sequential model of CNN+BiLSTM is able to have the lowest MSE and RMSE of 1.98 and 1.41 respectively while CNN+LSTM has MSE of 2.26 ad RMSE of 1.5.

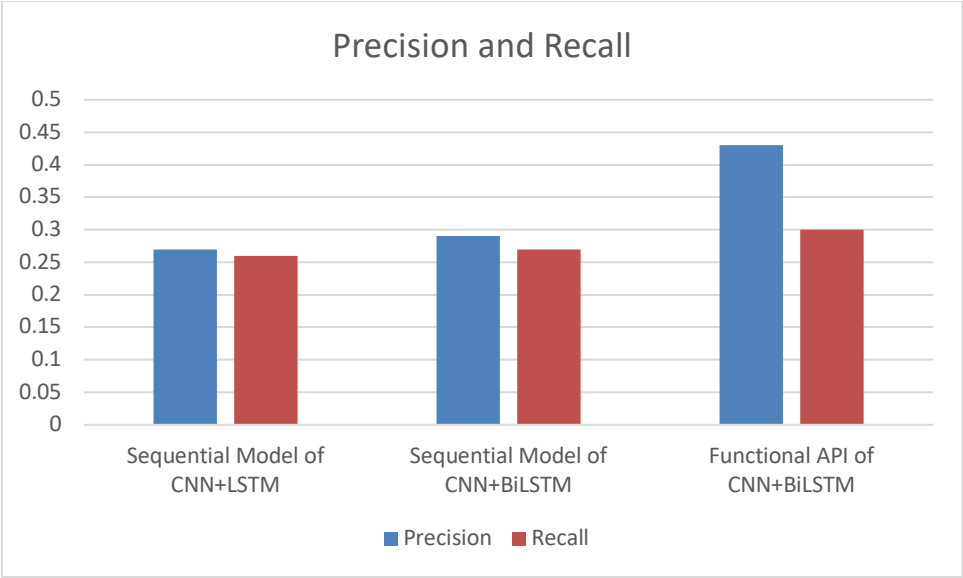

Figure 44: Neural Networks Precision and Recall

Figure (44) illustrates that the in terms of precision of and recall functional API of CNN+BiLSTM performed very well. The model has the highest count of positive class labelling, and the sequential model of CNN+BiLSTM follows it with 0.29 and 0.27 precision and recall, respectively.

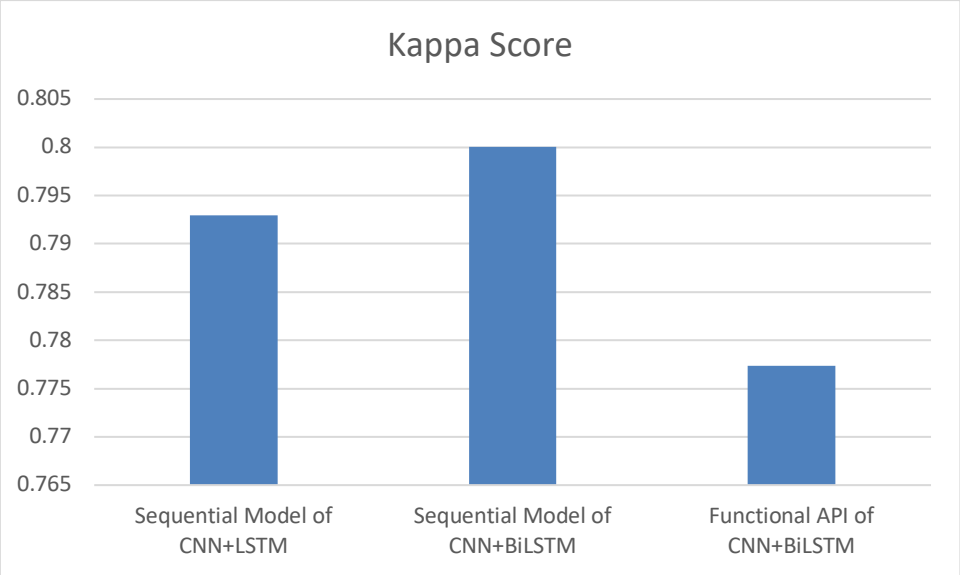

Figure 45: Kappa score in Neural Networks

Figure (45) illustrates that the sequential model of CNN+BiLSTM has the highest interrater agreement i.e. 0.80 while the sequential model of CNN+LSTM follows it with 0.79. it is observed that the sequential models gave a better agreement score instead of functional API with a 0.77 score.

4.2.3 Trial 3 – Transformers

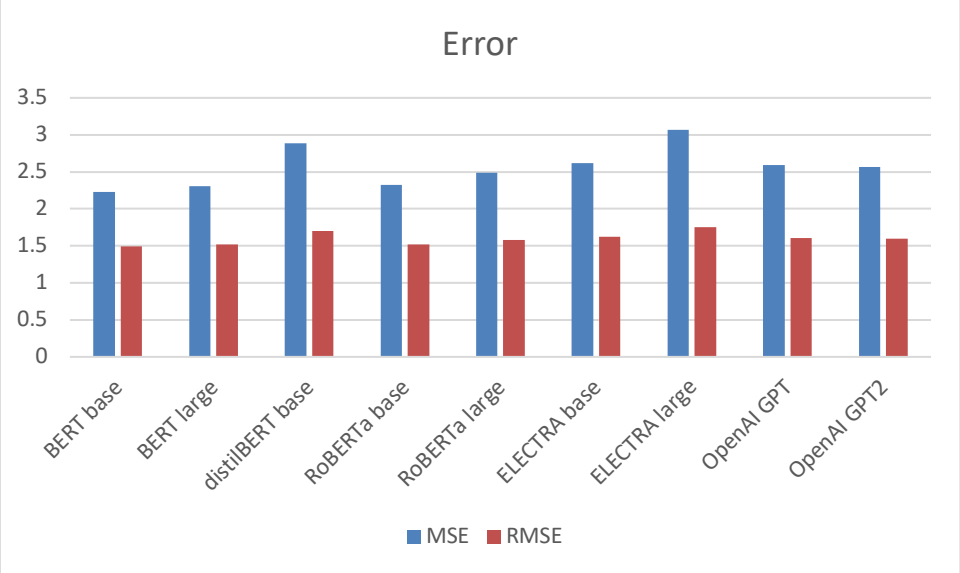

Figure 46: Transformers error count

Figure (46) shows the MSE and RMSE of the various transformer models which were trained on more than 12000 essays data set. BERT base was able to achieve the lowest MSE and RMSE out of all which is 2.23 and 1.49 respectively. The model follows score is BERT large and RoBERTa large with ME and RMSE of 2.31, 1.52 and 2.31, 1.52 respectively.

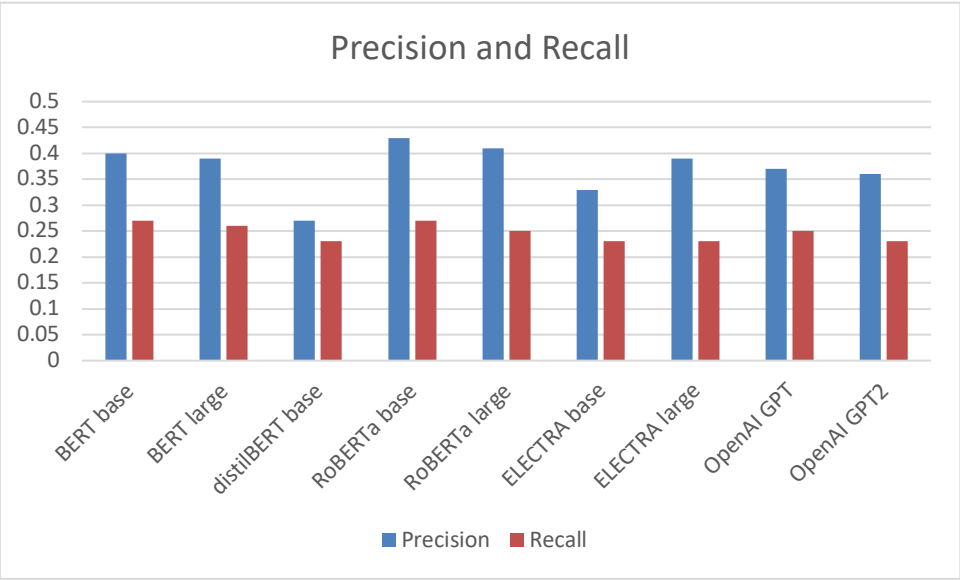

Figure 47: Transformers Precision and Recall

Figure (47) shows that the RoBERTA base has the highest precision of 0.43 followed by RobBERTa large with 0.41. it is also observed that approximately all the models have the same precision and recall with difference of **±0.7** and **±0.4** respectively.

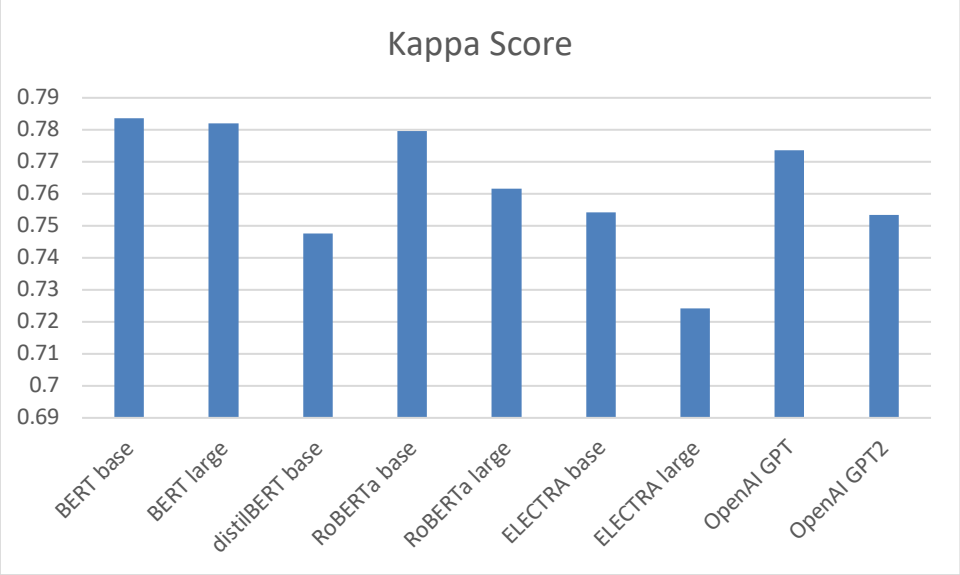

Figure 48: Kappa score in transforms

Figure (48) illustrates that approximately all the models have the kappa score in range of 0.7 with difference of **± 0.3.** BERT large and base has the same score of 0.78.

Chapter 5. Conclusion and Future Work

This section will cover the main points of the research. After that, the challenges that were faced during the project will be discussed. Finally, future work that can be carried out on in this project will be discussed.

5.1 Project summary

The summary of this project is to describe the comparison of different machine learning models using the; ASAP dataset that can efficiently and with minimal feature engineering provides the optimal results that can be considered without any doubt. Minimizing the large feature engineering section also results in a less computational cost that can sometimes take hours on personal computers.

Working on this project provides the good amount of experience of various machine learning models and how these models can be used in multi-class natural language problems. The parameters and doing normalization greatly affect the results of AI models. As our problem is a multi-class problem and mainly text classification, the simple loss function does not work. For that reason, we have used MSE and RMSE score, which provides the degree of prediction. Choosing different machine learning algorithms to train the model gives better insight into the best models for the provided NLP task.

Figure 49: Error Evaluation of All trials

From the figure (49) The deep learning models shows the good result in terms of MSE as the predictions show the lower mean square error in the models. The mean square error and root mean square error shown in trial 2 and 3 were taken after training models from test data.

5.2 Problems faced and lessons learned

Training and pre-processing of the NLP models require substantial computational requirements. Also, understanding the architectures of the models and the structures and models should have strong background knowledge of machine learning. Understanding the deep intuition of the models also requires extensive deep learning domain knowledge.

5.3 Future work

Because our thesis was centred on comparing machine learning models and producing outcomes based on fundamental feature engineering, the score or results presented in the thesis represent holistic grading of essays rather than domain-specific writing such as narrative or persuasive writing. Furthermore, it lacks the precise tuning of advanced neural networks, resulting in only equivalent outcomes. Fairness – whether subgroups of interest are treated equally by the scoring technique – is a crucial component of system performance to examine before real-world applications. Finally, the study's scope was limited to the topic of

how effectively automated essay scoring can mimic human raters' ratings across various sorts of prompts and scoring processes.

References

[1] E. B. Page, "The Imminence of... Grading Essays by Computer," *Phi Delta Kappa International,* vol. 47, no. 5, pp. 238-243, 1966.

[2] M. P. G. MAYO, "The Reliability of the Holistic Method," *Cuadernos de Filología Inglesa,* vol. 5, no. 1, pp. 51-62, 1996.

[3] B. L. Sevcikova, "Human versus Automated Essay Scoring: A Critical Review," *Arab World English Journal (AWEJ),* vol. 9, no. 2, pp. 157 -174, 2018.

[4] D. Charney, "The Validity of Using Holistic Scoring to Evaluate Writing: A Critical Overview," *Research in the Teaching of English,* vol. 18, no. 1, pp. 65-81, 1984.

[5] K. Taghipour and H. T. Ng, "A Neural Approach to Automated Essay Scoring," *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,* p. 1882–1891, 2016.

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," in *Neural Computation (1997)*, Cambridge, 1997.

[7] F. Dong, Y. Zhang and J. Yang, "Attention-based Recurrent Convolutional Neural," in *Conference on Computational Natural Language Learning (CoNLL 2017)*, Vancouver, 2017.

[8] R. T. Ionescu, M. Cozma, A. M. Butnaru and R. Tudor, "Automated essay scoring with string kernels and word embeddings," arXiv, https://arxiv.org/abs/1804.07954, 2018.

[9] Larkey and L. S., "Automatic essay grading using text categorization techniques," in *International ACM SIGIR conference on Research and development in information retrieval*, New York, 1998.

[10] Y. Attali and J. Burstein, " Automated Essay Scoring With e-rater® V.2," *The Journal of Technology, Learning, and Assessment,* vol. 4, no. 3, 2006.

[11] M. Kalz, H. Drachsler, J. v. Bruggen, H. Hummel and R. Koper, "Automated Essay Scoring: Applications to Educational Technology," *International Journal of Emerging Technologies in Learning (iJET),* vol. 3, no. 2, p. 24–28, 2008.

[12] D. T. K. Landauer and D. S. Dumais, "Latent semantic analysis," *Annual Review of Information Science and Technology,* vol. 38, no. 1, pp. 188-230, 2005.

[13] D. Alikaniotis, H. Yannakoudakis and M. Marek, "Automatic Text Scoring Using Neural Networks," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, 2016.

[14] T. Dasgupta, A. Naskar, L. Dey and R. Saha, "Augmenting Textual Qualitative Features in Deep Convolution Recurrent Neural Network for Automatic Essay Scoring," in *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, Melbourne, 2018.

[15] T. Kakkonen, N. Myller and E. Sutinen, "Applying Latent Dirichlet Allocation to Automatic Essay Grading," in *International Conference on Natural Language Processing*, Finland, 2006.

[16] M. M. Islam and A. S. L. Haque, "Automated Essay Scoring Using Generalized Latent Semantic Analysis," *Journal of Computers,* vol. 7, no. 3, pp. 616-626, 2012.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention Is All You Need," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, CA, 2017.

Appendix A Glossary

BiLSTM: Bi directional Long Short-term Memory

CNN: Convolutional Neural Network

NLP: Natural Language Processing

Appendix B Deployment/Installation Guide

For deploying the environment, the user needs to follow the below steps:
Step 1: Copy the jupyter notebook named All Transformers.ipynb to the computer
Step 2: After copying the notebook upload it to the google colab
Step 3: Set the google colab runtime to TPU
Step 4: After setting the TPU; link the environment to the google drive and use the Kaggle dataset.
Step 5: now run the cells as colab will use the latest stable releases of libraries.
Step 6: Repeat the same process on other 2 notebooks.

Appendix C User Manual

Step 1: Connect to the available TPUs with your Colab file

Step 2: Mount your file directory with Google Colab

Step 3: Read the dataset file with pandas and remove all the unnessory columns

Step 4: Normalize the essay scors with their own min and max score

Step 5: Remove any null rows

Step 6: Pre-processing of the essay by converting them to lower case, lematization with POS tagging and remove stop words.

Step 7: Tokenization with keras tokenizer

Step 8: Pad the essays because they are of multiple length

Step 9: Use GloVe pre-trained embedding to creat embedding matrix

Step 10: Model Creation with LSTM and CNN

Step 11: Splitting the dataset

Step 12: Model training with callbacks

Step 13: Predicting with unseen data

Step 14: Evaluation of performance with Kappa Score

Appendix D Student Information Sheet

Roll No	Name	Email Address (FC College)	Frequently Checked Email Address	Personal Cell Phone Number
21-11482	Jam Ayub	21-11482@formanite.fccollege.edu.pk	21-11482@formanite.fccollege.edu.pk	03054177096
21-11494	M. Talha Imran	21-11494@formanite.fccollege.edu.pk	92talhaimran@gmail.com	03084656565
21-11386	Umama Rashid	21-11386@formanite.fccollege.edu.pk	21-11386@formanite.fccollege.edu.pk	

Appendix E Plagiarism Free Certificate

This is to certify that, I am Jam Ayub S/D/o Ayub, group leader of FYP under registration no 21-11482 at Computer Science Department, Forman Christian College (A Chartered University), Lahore. I declare that my Final year project report is checked by my supervisor and the similarity index is 19% that is less than 20%, an acceptable limit by HEC. Report is attached herewith as Appendix F. To the best of my knowledge and belief, the report contains no material previously published or written by another person except where due reference is made in the report itself.

Date: 30/01/2022 Name of Group Leader: Jam Ayyub Signature: _____

Name of Supervisor: Ali Faheem Co-Supervisor (if any): Dr. M. Haroon Shakeel
Designation: Lecturer Designation: Lecturer
Signature: _____ Signature: _____

Senior Project Management Committee Representative: _____
Signature: _____

Appendix F Plagiarism Report

srs revised

ORIGINALITY REPORT

19%	11%	9%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	medium.com Internet Source	3%
2	Submitted to Higher Education Commission Pakistan Student Paper	2%
3	www.ijcai.org Internet Source	1%
4	Mafizur Rahman, Maryam Mehzabin Zahin, Linta Islam. "Effective Prediction On Heart Disease: Anticipating Heart Disease Using Data Mining Techniques", 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT), 2019 Publication	1%
5	Jin, Cancan, and Ben He. "Utilizing Latent Semantic Word Representations for Automated Essay Scoring", 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing	1%

	and Communications and Its Associated Workshops (UIC-ATC-ScalCom), 2015. Publication	
6	Submitted to Aston University Student Paper	1 %
7	Majdi Beseiso, Saleh Alzahrani. "An Empirical Analysis of BERT Embedding for Automated Essay Scoring", International Journal of Advanced Computer Science and Applications, 2020 Publication	<1 %
8	jitm.ut.ac.ir Internet Source	<1 %
9	Submitted to Athens University of Economics and Business Student Paper	<1 %
10	Submitted to South Bank University Student Paper	<1 %
11	Submitted to University of Sydney Student Paper	<1 %
12	Submitted to Ain Shams University Student Paper	<1 %
13	www.techtarget.com Internet Source	<1 %
14	Keshav Kumar, Sadia Khanam, Md Mahbub Islam Bhuiyan, Mohammad Reza Chalak	<1 %

	Qazani et al. "SpinalXNet: Transfer Learning with Modified Fully Connected Layer for X-Ray Image Classification", 2021 IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE), 2021 Publication	
15	Submitted to Birkbeck College Student Paper	<1 %
16	Submitted to University of Bradford Student Paper	<1 %
17	Submitted to Harrisburg University of Science and Technology Student Paper	<1 %
18	Submitted to Queen Mary and Westfield College Student Paper	<1 %
19	Submitted to National Research University Higher School of Economics Student Paper	<1 %
20	Şükrü Mustafa Kaya, Atakan Erdem, Ali Güneş. "A Smart Data Pre-Processing Approach to Effective Management of Big Health Data in IoT Edge", Smart Homecare Technology and TeleHealth, 2021 Publication	<1 %
21	Submitted to University of Northampton Student Paper	<1 %

22	Spraha Kumawat, Inna Yadav, Nisha Pahal, Deepti Goel. "Sentiment Analysis Using Language Models: A Study", 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2021 Publication	<1 %
23	digitalcommons.wpi.edu Internet Source	<1 %
24	Submitted to British University in Egypt Student Paper	<1 %
25	Submitted to University of Technology, Sydney Student Paper	<1 %
26	Submitted to The Robert Gordon University Student Paper	<1 %
27	dokumen.pub Internet Source	<1 %
28	Submitted to University of Florida Student Paper	<1 %
29	Submitted to University of Queensland Student Paper	<1 %
30	Xiaobing Wang, Ge Li, Chunyi Li, Liang Zhao, Xinfeng Shu. "Chapter 11 Automatic Generation of Specification from Natural	<1 %

	Language Based on Temporal Logic", Springer Science and Business Media LLC, 2021 Publication	
31	arxiv.org Internet Source	<1 %
32	curve.coventry.ac.uk Internet Source	<1 %
33	educationdocbox.com Internet Source	<1 %
34	lib.dr.iastate.edu Internet Source	<1 %
35	www.modernscientificpress.com Internet Source	<1 %
36	Submitted to Edith Cowan University Student Paper	<1 %
37	Pooja Vinayak Kamat, Rekha Sugandhi, Satish Kumar. "Deep learning-based anomaly-onset aware remaining useful life estimation of bearings", PeerJ Computer Science, 2021 Publication	<1 %
38	mobt3ath.com Internet Source	<1 %
39	Submitted to Coventry University Student Paper	<1 %
40	Submitted to University of Hertfordshire Student Paper	

		<1 %
41	Submitted to University of Durham Student Paper	<1 %
42	"Intelligent Data Communication Technologies and Internet of Things", Springer Science and Business Media LLC, 2021 Publication	<1 %
43	Submitted to CSU, San Jose State University Student Paper	<1 %
44	Submitted to Institute of Technology Blanchardstown Student Paper	<1 %
45	Siyuan Zhao, Yaqiong Zhang, Xiaolu Xiong, Anthony Botelho, Neil Heffernan. "A Memory-Augmented Neural Model for Automated Grading", Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale - L@S '17, 2017 Publication	<1 %
46	Submitted to University of Warwick Student Paper	<1 %
47	Submitted to Imperial College of Science, Technology and Medicine Student Paper	<1 %

48	Li Yang, Ying Li, Jin Wang, Zhuo Tang. "Post Text Processing of Chinese Speech Recognition Based on Bidirectional LSTM Networks and CRF", Electronics, 2019 Publication	<1 %
49	dergipark.org.tr Internet Source	<1 %
50	Submitted to Liverpool John Moores University Student Paper	<1 %
51	Miss Dhara N. Darji, Satyen M. Parikh, Hiral R. Patel. "Chapter 39 Sentiment Analysis of Unstructured Data Using Spark for Predicting Stock Market Price Movement", Springer Science and Business Media LLC, 2022 Publication	<1 %
52	Rosario Catelli, Francesco Gargiulo, Emanuele Damiano, Massimo Esposito, Giuseppe De Pietro. "Clinical de-identification using sub-document analysis and ELECTRA", 2021 IEEE International Conference on Digital Health (ICDH), 2021 Publication	<1 %
53	Submitted to University of Hong Kong Student Paper	<1 %
54	towardsdatascience.com Internet Source	<1 %

55	Bronwyn Woods, David Adamson, Shayne Miel, Elijah Mayfield. "Formative Essay Feedback Using Predictive Scoring Models", Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17, 2017 Publication	<1 %
56	Marvin Chandra Wijaya. "Automatic Short Answer Grading System in Indonesian Language Using BERT Machine Learning", Revue d'Intelligence Artificielle, 2021 Publication	<1 %
57	Masaki Uto. "A review of deep-neural automated essay scoring models", Behaviormetrika, 2021 Publication	<1 %
58	bradscholars.brad.ac.uk Internet Source	<1 %
59	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %
60	ebin.pub Internet Source	<1 %
61	Nithin Valiyaveedu, Sangeetha Jamal, Roshan Reju, Vysakh Murali, Nithin K M. "Survey and Analysis on AI Based Phishing Detection Techniques", 2021 International Conference	<1 %

	on Communication, Control and Information Sciences (ICCISc), 2021 Publication	
62	github.com Internet Source	<1 %
63	journal.auric.kr Internet Source	<1 %
64	www.diva-portal.org Internet Source	<1 %
65	Adeem Akhtar, Taha Ahmad, Nosheen Sabahat, Sidra Minhas. "IoT Based Home Automation System Using ThingSpeak", 2019 International Conference on Computing, Electronics & Communications Engineering (iCCECE), 2019 Publication	<1 %
66	Nikhil Yadav, Omkar Kudale, Aditi Rao, Srishti Gupta, Ajitkumar Shitole. "Chapter 51 Twitter Sentiment Analysis Using Supervised Machine Learning", Springer Science and Business Media LLC, 2021 Publication	<1 %
67	Submitted to Universiti Teknologi Malaysia Student Paper	<1 %
68	api.intechopen.com Internet Source	<1 %

69	argon.ess.washington.edu Internet Source	<1 %
70	edepot.wur.nl Internet Source	<1 %
71	link.springer.com Internet Source	<1 %
72	peerj.com Internet Source	<1 %
73	"Software Engineering Perspectives in Intelligent Systems", Springer Science and Business Media LLC, 2020 Publication	<1 %
74	Submitted to Bournemouth University Student Paper	<1 %
75	Submitted to East Carolina University Student Paper	<1 %
76	James Graham, Janusz A. Starzyk. "Transitioning from motivated to cognitive agent model", 2013 IEEE Symposium on Computational Intelligence for Human-like Intelligence (CIHLI), 2013 Publication	<1 %
77	Meryem Souaidi, Mohamed El Ansari. "Multi-scale analysis of ulcer disease detection from WCE images", IET Image Processing, 2019 Publication	<1 %

78	Zhaohong Deng, Yizhang Jiang, Hisao Ishibuchi, Kup-Sze Choi, Shitong Wang. "Enhanced Knowledge-Leverage-Based TSK Fuzzy System Modeling for Inductive Transfer Learning", ACM Transactions on Intelligent Systems and Technology, 2016 Publication	<1 %
79	arrow.tudublin.ie Internet Source	<1 %
80	ceur-ws.org Internet Source	<1 %
81	esource.dbs.ie Internet Source	<1 %
82	www.uni-tuebingen.de Internet Source	<1 %
83	Guoxi Liang, Byung-Won On, Dongwon Jeong, Hyun-Chul Kim, Gyu Choi. "Automated Essay Scoring: A Siamese Bidirectional LSTM Neural Network Architecture", Symmetry, 2018 Publication	<1 %
84	Islam, Md. Monjurul, and A. S. M. Latiful Hoque. "Automated Essay Scoring Using Generalized Latent Semantic Analysis", Journal of Computers, 2012. Publication	<1 %

85	Odunayo Esther Oduntan, Stephen Olatunde Olabiyisi, Ibrahim Adepoju Adeyanju, Elijah Olusayo Omidiora. "A modified principal component analysis approach to automated essay-type grading", 2016 Future Technologies Conference (FTC), 2016 Publication	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		