Automated Essay Scoring: Methodologies, Limitations, and Knowledge Transfer

Lorenzo D'Antoni
lorenzo.dantoni@studenti.unipd.it

Alessandro Canel
alessandro.canel@studenti.unipd.it

Davide Bassan
davide.bassan.1@studenti.unipd.it

Hannaneh Kalantary
hannaneh.kalantary@studenti.unipd.it

Hooman Sabzi
hooman.sabzi@studenti.unipd.it

Abstract

This report investigates the intricate landscape of Automated Essay Scoring (AES), a computer-based system tasked with evaluating student essays. Despite notable computational and algorithmic advancements, existing AES models fall short in terms of performance and generalizability. The study critically assesses various AES methodologies, encompassing classical machine-learning and contemporary deep-learning techniques. It underscores the limitations of current models, particularly their overreliance on syntactic features while neglecting content quality, cohesion, coherence, and domain-specific knowledge. The research aims to test the effectiveness of these AES methodologies and to explore the feasibility of model knowledge transfer across datasets with distinct scoring systems. The findings emphasize the need for AES models that can fully comprehend the intricacies of essay content, thereby enhancing their practicality in educational settings.

1. Introduction

Automated essay scoring (AES) evaluates student responses using various features, addressing the challenges of manual grading, which is time-consuming and unreliable. While automated systems efficiently handle multiple-choice questions, assessing essays remains difficult. Most research has focused on syntactic-based assessments rather than content-based evaluations.

This report explores the nuanced and complex field of AES, where no single approach consistently outperforms others. Despite improvements in state-of-the-art models, they still lack the performance and generalizability needed for real-world applications. Our work focuses on English-language essays.

Our objectives were to test different approaches to AES, including classical machine learning and deep learning methods, and to determine if knowledge transfer occurs between models trained on datasets with different score ranges and criteria.

Section 2 explores the most relevant works and systems in the AES literature. It provides details on the features used by the models, the most commonly used evaluation metrics, the techniques and approaches employed, the design of cutting-edge models, and the limitations and challenges faced in this field. Section 3 describes the data used for our experiments, its sources, and any preprocessing applied. Section 4 outlines our objectives and approaches. Section 5 presents our experiments, results, areas for improvement, and encountered challenges. Finally, Section 6 summarizes our findings and suggests ideas for future research.

2. Related Work

Research on Automatic Essay Scoring (AES) began in 1966 with the development of the *Project Essay Grader* (PEG) [2], which evaluated writing characteristics like grammar, diction, and construction.

In 1999, the *Intelligent Essay Assessor* (IEA) [8] was introduced, using latent semantic analysis to evaluate content and score essays. Subsequent systems like *E-rater* [11] in 2002, *Intellimetric* [13] in 2006, and the *Bayesian Essay Test Scoring System* (BESTY) [14] in 2002, employed NLP techniques to enhance scoring accuracy and reliability.

Since 2014, AES systems have increasingly adopted deep learning techniques, with models developed by [7] leveraging syntactic and semantic features for improved performance.

In the last decade, AES systems have advanced to score essays based on syntax and semantics, using three types of features: **statistical-based** features [4] [10] (word frequency, sentence length, etc.), **style-based** (syntax) features [5] [6] (grammar, punctuation, sentence complexity, stylis-

tic elements, etc.), and **content-based** features [7] (word embeddings).

AES systems are typically evaluated using three metrics: **Quadratic Weighted Kappa** QWK (measures the agreement between two raters, adjusting for the possibility of agreement occurring by chance), **Mean Absolute Error** MAE, and **Pearson Correlation Coefficient** PCC (assesses the linear correlation between the predicted scores and the actual scores).

Techniques used in automated essay grading fall into four groups, all based on supervised learning: **regression** (Shortest Path, Ridge Regression), **classification** (SVM, Random Forest, XGBoost), **neural networks** (Hierarchical CNN, Attention-based CNN+LSTM, Bi-GRU, Bi-LSTM with word2vec embeddings, Convolution RNN), and **ontology-based** approaches (semantic similarity measures).

2.1. Cutting-edge models

Cutting-edge models [15] combine statistical indicators, TF-IDF features, and deep learning outputs, typically from BERT family models.

The essay undergoes preprocessing steps like HTML tag removal, punctuation removal, spelling error detection, and contraction expansion. Linguistic features such as paragraph length, sentence count, and word count are then extracted.

These features, along with BERT model embeddings and TF-IDF features, are fed into a regression or classification model. Often, an ensemble model, such as those in gradient boosting frameworks, is used to enhance scoring robustness and reliability.

2.2. Challenges and limitations

Despite advancements in AES, current models face significant challenges. They do not effectively evaluate essay content relevance to prompts or adequately address cohesion and coherence, despite some efforts using latent semantic analysis.

Many approaches focus on improving Quadratic Weighted Kappa (QWK) scores, but this metric does not sufficiently assess feature extraction or detect irrelevant answers, leading to potentially inaccurate assessments. Models, including deep learning systems, often fail to recognize and appropriately score irrelevant or adversarial content, allowing students to exploit the system by including prompt vocabulary or shell language (superficial or generic content that is inserted into an essay to give the appearance of relevance to the prompt without actually providing meaningful information) to manipulate scores.

Metrics	Values
3	6280
2	4723
4	3926
1	1252
5	970
6	156

Table 1. Distribution of essay scores in the dataset provided by [1] (*Learning Agency Lab - Automated Essay Scoring 2.0*).

3. Datasets

We encountered difficulties in finding a dataset with sufficient data for training a deep learning model that wasn't severely unbalanced. Numerous small datasets are available online, each with its scoring range and rules, such as the *Cambridge Learner Corpus-First Certificate* in English exam (CLC-FCE) [18], which contains 1244 essays across 10 prompts.

The only datasets with adequate data were from two AES competitions hosted on Kaggle: one held in 2012 [17] and another in 2024 [1]). The latter competition provided 17307 student-written argumentative essays, each scored on a scale of 1 to 6. Despite the richness of data, this dataset suffered from severe imbalance, as depicted in Table 1, yet it remained the best available option.

To address the imbalance issue, we considered using the Synthetic Minority Over-sampling Technique (SMOTE) algorithm. However, SMOTE failed to generate logically structured essays that capture the semantic nuances and coherence of genuine ones when applied to our dataset.

In our quest for datasets containing student-written essays from an English certification program, quality resources were scarce. Hence, we chose to utilize the *IELTS Writing Scored Essays Dataset* [9], as employed by [16]. This dataset aggregates exemplar essays from the writing component of the IELTS examination.

Despite including additional columns like examiner comments, much of the data is missing. Scores range from 1 to 9, with 0.5 increments, but, like other datasets, a significant class imbalance exists. To split the data into training, validation, and test sets, a minimum of three essays per score category was needed, necessitating the removal of some scores and associated essays.

4. Method

The aim is to evaluate NLP pre-trained models' efficacy in AES without extra syntactic features. In this way, we can assess to which extent the NLP models contribute to the final AES score.

Our approach involved utilizing a pre-trained NLP model and refining it with the dataset provided by [1]. Our

Metrics	Values
6.0	264
7.0	254
6.5	250
5.5	176
7.5	138
8.0	137
5.0	104
9.0	37
8.5	35
4.5	21
4.0	11
3.5	5
3.0	2
1.0	1

Table 2. Distribution of essay scores in the*IELTS Writing Scored Essays Dataset* [9].

objective was to leverage the knowledge gained from this model to evaluate the essays written by students attending the IELTS.

With limited IELTS essay data, we aimed to determine if knowledge transfer is feasible across datasets and systems with varied grading ranges.

We selected BERT models due to their success in the Kaggle competition [1].

In the next section, we detail the experiments conducted, the architectures employed, the metrics utilized, the results obtained, and potential strategies for improving performance.

5. Experiments

5.1. Experimental Setup and Environment

We ran the experiments both on the Kaggle workspace which uses the NVIDIA Tesla P100 GPU and on a local workstation: Ubuntu 20.04.6 LTS, NVIDIA GeForce RTX 3060 12GB GPU, AMD Ryzen 5 3600 CPU, and DDR4 16GB of RAM. We used *tensorflow* 2.15.0, *transformers* 4.41.2.

5.2. Baseline

We initially developed a basic k-NN model, serving as our baseline, which utilized only the text and the score as features. This preliminary model achieved a validation accuracy of 0.35.

To enhance its performance, we expanded the feature set to include the essay's length and sentiment polarity. This integration led to a notable improvement, raising the validation accuracy to 0.48. However, as we'll explain later, this inclusion introduces certain biases into the model.

Subsequently, we tried to optimize the number of neighbors (k) in the k-NN algorithm. Employing cross-validation, we determined the optimal k value to be 27, resulting in an enhanced validation accuracy of 0.55.

In our pursuit of further improvement, we introduced additional features such as word count, unique word count, average word length, sentence count, average sentence length, and lexical diversity. However, their inclusion led to a marginal decrease in validation accuracy, dropping to 0.51. Despite applying `GridSearchCV` from the *scikit-learn* library for hyperparameter tuning, the validation accuracy remained stagnant at 0.50.

To leverage the strength of ensemble methods, we assembled a trio of k-NN models with varying numbers of neighbors (10, 20, 30). This ensemble model yielded a substantial boost in validation accuracy, reaching 0.57.

5.3. Strategies for Enhancing Model Performance

In the Kaggle competition [1], most notebooks used *PyTorch* and Hugging Face's *Transformers* library for easy access to pre-trained NLP models. Given that the *Transformers* library supports interoperability with *PyTorch*, *TensorFlow*, and *JAX*, we chose *TensorFlow* for our experiments. This decision was also influenced by [16] (GitHub repository) that used TensorFlow for grading IELTS essays. However, we identified structural and logical errors in that repository, which we aimed to address.

The code in [16] split the data into training and validation sets but did not set aside a test set, which is crucial for unbiased performance evaluation on unseen data.

We processed the input text into tokens suitable for BERT, using *WordPiece* tokenization. These tokens were converted into embeddings, with positional embeddings added to preserve word order. The tokenizer's `max_length` parameter establishes the maximum length for tokenized sequences. Setting `truncation=True` truncates longer sequences, and `padding=max_length` ensures all sequences are padded to the specified length. Padding and truncation create uniform-length sequences, essential for efficient GPU computation. However, the repository set padding to`True` instead of `max_length`, potentially impacting model performance.

The tokenizer's output includes:

- `input_ids`: a tensor of token IDs representing the tokenized text.
- `attention_mask`: a tensor indicating which tokens should be attended to by the model (1 for real tokens, 0 for padding tokens), though this was not utilized in the repository.
- `token_type_ids`: a tensor distinguishing between text segments (relevant for tasks like question answering).

Initially, we attempted to use the DeBERTa model, known for its advanced techniques and regarded as a powerful BERT-based model. However, Hugging Face did not officially support DeBERTa V3 at that time, and only the larger versions of DeBERTa V2 were available, which exceeded our hardware capabilities. Consequently, we reverted to the BERT base model.

Using BERT, the suggested maximum input sequence length is 512 tokens. Beyond this limit, BERT's performance begins to degrade. Many essays exceeded this limit, resulting in significant information loss (Figure 1) . Although newer models can handle longer sequences better, our computational resources were insufficient to use them effectively.

5.3.1 Classification approach

Our initial approach was to tackle the problem as a classification task. We augmented the BERT model with a classification head, consisting of two fully connected layers with 128 and 64 neurons, respectively, using ReLU activation. This was followed by a final layer with a single neuron and softmax activation. To mitigate overfitting, we added dropout layers (dropout rate of 0.3). We used the Adam optimizer with an initial learning rate of 5e-4.

The highest accuracy achieved on the test set was 0.57. While this may seem low, it is consistent with other models using accuracy as the evaluation metric for AES, as noted in this systematic literature review [12].

We realized that essay scores follow an inherent order. Misclassifying a score of 3 as 4 is less severe than predicting a score of 1. However, treating the task as a classification problem means each misclassification carries equal weight, ignoring the distance from the true score.

5.3.2 Ordinal Regression

Ordinal regression predicts an ordinal dependent variable, unlike typical regression with a continuous target variable. It captures the inherent order in a discrete, ordered set of values.

We found this approach in a notebook [3] submitted for the previously mentioned Kaggle Competition [1]. The authors used *KerasNLP*, a NLP library compatible with *TensorFlow*, *JAX*, and *PyTorch*. Given the simplicity and effectiveness of their code, we used their results directly instead of reimplementing the idea in *TensorFlow*.

They transformed essay scores into an ordinal matrix for loss calculation. For example, a score of 3 out of 6 is represented as $[1, 1, 1, 0, 0, 0]$.

They tokenized the input texts and employed a BERT model augmented with a fully connected layer of 64 neurons (ReLU activation) and a final layer with 6 neurons (sigmoid activation), producing an output vector with values ranging from 0 to 1.

We used the Adam optimizer with an initial learning rate of 5e-6 and binary cross-entropy loss. For model evaluation, we applied the Quadratic Weighted Kappa (QWK) metric, converting the output vectors to a single integer score.

On the test set, the highest QWK achieved was approximately 0.73, a commendable score considering the current state of the art.

The ordinal regression model achieved a good QWK value but faced issues when fine-tuning the small IELTS dataset. We couldn't insert scores in the range of $[1, 9]$ with 0.5 increments into the model. Transforming the IELTS scores to a $[1, 6]$ range with 1 increments was feasible, but the small and unbalanced dataset led to poor performance. Moreover, predicting IELTS scores in the $[1, 6]$ range is impractical since they don't use that scale.

5.3.3 Regression: hyperbolic tangent approach

We attempted to scale the scores from the $[1, 6]$ range to the $[-1, 1]$ range, using the tanh activation function in the final layer to predict values in this range. The aim was to achieve a low mean squared error (MSE) on the original data after rescaling them back to $[1, 6]$. We maintained the same head architecture used for classification, but added a fully-connected layer with 256 neurons. Dropout regularization (0.3 rate) was applied after each layer, and the Adam optimizer with a learning rate of 3e-5 was used for training.

However, the best MSE on the test set was 0.1, which is too high for scaled data. Rescaling the scores back to the original scale resulted in an MSE of around 1. This outcome was expected since the errors in the $[-1, 1]$ range were small compared to the original scale due to the squaring of differences in the MSE metric.

The same issue encountered with the ordinal regression approach also affected regression models using the tanh activation function. Despite different target ranges, the problem of scaling the scores persisted. Due to suboptimal performance on the initial dataset, we did not fine-tune the model on the IELTS dataset.

5.3.4 Regression: linear approach

Following the approach used in [16], we experimented with a linear activation function in the final layer of the regression head on the BERT model. We had reservations about this method due to its lack of domain knowledge of the scores and potential issues with exploding gradients and unstable learning. The results from the repository were also not promising.

Despite these concerns, a linear activation function avoids the need to scale scores and allows for evaluating knowledge transfer between datasets. However, our initial

Metrics	Kaggle	IELTS
Test MSE	0.63	0.92
Test MAE	0.50	0.69
Test R^2	0.56	0.21
Test RMSE	0.66	0.84

Table 3. Model Performance Metrics: Mean Squared Error (MSE), MAE (Mean Absolute Error), R^2, and Root Mean Squared Error (RMSE) based on the Kaggle [1] and IELTS [9] test datasets.

experiments showed that training was unstable and slow. To address this, we increased the batch size from 16 to 32, expanded the complexity of the regression head to six fully connected layers with 512, 256, 256, 128, 64, and 1 neurons (using ReLU activation), incorporated batch normalization after each layer in the regression head, implemented dropout with a rate of 0.3 after each layer, used AdamW with an initial learning rate of 3e-5 and a weight decays of 1e-5 and increased the number of epochs to 200.

We used Mean Squared Error (MSE) as the loss function and Mean Absolute Error (MAE), R^2 score, and Root Mean Squared Error (RMSE) as performance metrics.

Training the model for 200 epochs resulted in a smooth convergence process (Figure 2). The results are presented in Table 3, where the Mean Absolute Error (MAE) decreased from 0.71 to 0.51 compared to the benchmark in [16]. However, Figure 3 reveals that the model struggles with predicting classes that are not well-represented in the data. This is further confirmed by the distribution of residuals in Figure 4, which suggests the presence of large errors.

When fine-tuning the model with the IELTS dataset [9], we observed no noticeable transfer of knowledge (Table 3). Freezing and unfreezing the weights of various layers in the regression head did not improve performance. We also increased the learning rate to 3e-4 and used a learning rate scheduler (`ReduceLROnPlateau`). This scheduler reduced the learning rate by a factor of *f* if there was no improvement in the validation loss for *p* epochs until a minimum learning rate was reached. Despite testing various hyperparameter values, we did not achieve significant improvement (Figure 5).

5.4. Final considerations

We experimented with various approaches to the AES problem to gain a deeper understanding, despite not using the latest cutting-edge models.

Initially, we implemented a baseline model based on the k-NN algorithm and attempted to improve it by incorporating various features and optimization strategies. We then shifted our focus to deep learning models.

Across all methods, we struggled to transfer knowledge effectively from one dataset with a specific score range to another. The most promising approach, excluding the fine-tuning of the IELTS dataset, was the ordinal regression model. Encoding the scores as vectors of zeros and ones and allowing the model to predict each entry individually proved efficient and effective for enhancing the model's learning of scores.

Overall, our results were below state-of-the-art benchmarks due to several factors:

- **Unbalanced Kaggle Competition Dataset**: the dataset, while substantial, was highly unbalanced. Our attempts at data augmentation did not yield satisfactory results, and we could not find another dataset with a comparable amount of data.

- **Small IELTS Dataset**: the IELTS dataset [9] was too small to effectively train a deep learning model. Even with a pre-trained model on a similar task and freezing some layers, the dataset's imbalance and lack of associated essays for all scores undermined learning.

- **BERT Model Input Length**: the maximum input length of the BERT model (512 tokens) resulted in the truncation of some essays, leading to the loss of valuable data.

- **Limited Computational Resources**: we lacked the computational resources to train larger, state-of-the-art NLP models, which limited our choice of models and their effectiveness.

- **Focus on Semantics Over Syntax**: by applying regression layers to the representation of the `[CLS]` token from BERT, we focused on the semantic aspects of the essays but did not adequately consider their syntactic structure.

The code from [16] attempted to concatenate the output of the BERT model with numerical values, such as essay length and the number of writing exercises (1 or 2) in the IELTS writing test. While this strategy slightly improved performance, we disagree with its implementation. Longer essays may tend to receive higher scores (Figure 6), but this should be an indirect consequence rather than a direct parameter for evaluation. Once the minimum length requirement is met, essay length should not be a primary criterion for assessment. Relying on essay length can introduce bias and undermine the model's generalizability, allowing students to achieve higher scores by writing longer essays without necessarily improving text quality.

Despite this, as mentioned in Section 2, the underlying idea is sound. The best results were obtained by combining syntactical, statistical, and style features with the output of an NLP model that focuses on semantics.

6. Conclusion

In this project, we have studied automated essay scoring (AES), examining various approaches. Despite advancements in algorithms and hardware, AES remains challenging due to several limitations.

We investigated several methods, including classical machine learning algorithms such as k-NN and deep learning models. In the realm of deep learning, we experimented with classification, ordinal regression, regression with tanh activation, and regression with linear activation. However, limitations in data quality, dataset size, model selection, and computational resources hindered performance.

Our research revealed that transfer learning between datasets with different score ranges and criteria is highly constrained. To obtain better results, we need to integrate not only the output of an NLP deep model but also various syntactic and statistical features from the text into a larger deep-learning model.

Nevertheless, developing effective and reliable AES models is complex. Current strategies struggle to evaluate content relevance to prompts and address cohesion and coherence between sentences and paragraphs.

In our view, future research should prioritize addressing these limitations rather than concentrating only on syntactic features. This can prevent syntactic biases and improve model generalizability. Additionally, exploring methods to provide constructive feedback to students can enhance educational value.

References

[1] The Learning Agency Lab · Featured Code Competition · a month to go Learning Agency Lab. Learning agency lab - automated essay scoring 2.0, 2024. Accessed on June 3, 2024.

[2] H. B. Ajay, P. I. Tillett, and E. B. Page. Analysis of essays by computer (aec-ii) final report. Technical report, National Center for Educational Research and Development, Washington, 1973.

[3] Awsaf. Aes 2.0: Kerasnlp starter, 2024. Accessed on June 3, 2024.

[4] Juan O. Contreras, Shahera M. Hilles, and Zulkifli B. Abubakar. Automated essay scoring with ontology based on text mining and nltk tools. In *2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE)*, pages 1–6, 2018.

[5] Ronan Cummins, Mian Zhang, and Ted Briscoe. Constrained multi-task learning for automated essay scoring. In *Proceedings of the Association for Computational Linguistics*, August 2016.

[6] S.M. Darwish and S.K. Mohamed. Automated essay evaluation based on fusion of fuzzy ontology and latent semantic analysis. In Aboul Ella Hassanien, Ahmad Taher Azar, Tarek Gaber, Rekha F. Bhatnagar, and Mohamed F. Tolba, editors, *The International Conference on Advanced Machine Learning Technologies and Applications*, 2020.

[7] Feng Dong, Yue Zhang, and Jie Yang. Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 153–162, 2017.

[8] Peter W. Foltz, Darrell Laham, and Thomas K. Landauer. The intelligent essay assessor: Applications to educational technology. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 1(2), 1999.

[9] ibrahimmazlum. Ielts writing scored essays dataset, 2023. Accessed on June 3, 2024.

[10] Yaman Kumar, Saket Aggarwal, Debanjan Mahata, Rajiv Ratn Shah, Ponnurangam Kumaraguru, and Roger Zimmermann. Get it scored using autosas—an automated system for scoring short answers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9662–9669, July 2019.

[11] Donald E. Powers, Jill C. Burstein, Martin Chodorow, Martha E. Fowles, and Karen Kukich. Stumping e-rater: Challenging the validity of automated essay scoring. *Computers in Human Behavior*, 18(2):103–134, 2002.

[12] D. Ramesh and S.K. Sanampudi. An automated essay scoring systems: a systematic literature review. *Artificial Intelligence Review*, 55:2495–2527, 2022. See Section 3.4.

[13] Lawrence M. Rudner, Valerie Garcia, and Charlotte Welch. An evaluation of intellimetric™ essay scoring system. *The Journal of Technology, Learning and Assessment*, 4(4), 2006.

[14] Lawrence M. Rudner and Tian Liang. Automated essay scoring using bayes' theorem. *The Journal of Technology, Learning and Assessment*, 1(2), 2002.

[15] Ryenhails. Deberta+lgbm with detailed code comments, 2024. Accessed on June 3, 2024.

[16] Aleksandr Shishkov. Logisx/deepessay, 2023. Accessed on June 3, 2024.

[17] The William and Flora Hewlett Foundation (Hewlett). The hewlett foundation: Automated essay scoring, 2012. Accessed on June 3, 2024.

[18] Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, 2011.

Appendix

A. Author contributions and working plan

- **Literature Review** (Lorenzo D'Antoni): conducted a comprehensive review of the Automated Essay Scoring (AES) literature, detailing influential models, key features, prevalent evaluation metrics, and current methodologies. Also identified the field's primary challenges and limitations.
- **Data Analysis** (Hannaneh Kalantary, Hooman Sabzi): performed the analysis of the datasets, identifying patterns, assessing data quality, and creating visual representations for enhanced understanding.
- **Baseline Model** (Alessandro Canel): developed a baseline essay grading model serving as a benchmark for subsequent advanced models.
- **Experiments** (Davide Bassan, Lorenzo D'Antoni): designed and implemented all the approaches described in the report. In particular:
  - Lorenzo D'Antoni: conducted preliminary experiments to evaluate various approaches, architectures, and hyperparameters on the Kaggle platform.
  - Davide Bassan and Lorenzo D'Antoni: collaboratively worked on fine-tuning models using the IELTS dataset.
  - Davide Bassan: managed extensive experimental runs and model training on his local setup, and produced the corresponding figures and charts.
  - Davide Bassan: prepared the notebook with the code for submission.
- **Report writing** (Lorenzo D'Antoni): synthesized the research findings, compared experimental outcomes, and articulated conclusions and prospective research avenues.

B. Figures and Charts

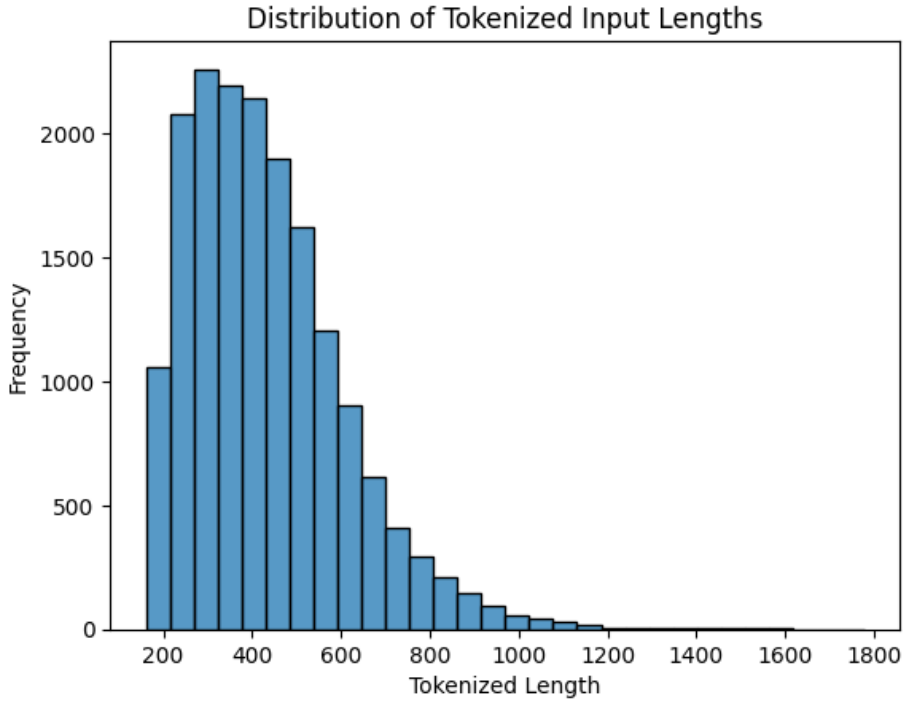

Figure 1. Histogram showing the distribution of tokenized input lengths in the dataset. Most tokenized inputs fall between 200 and 600 tokens, with a gradual decline in frequency for longer inputs.

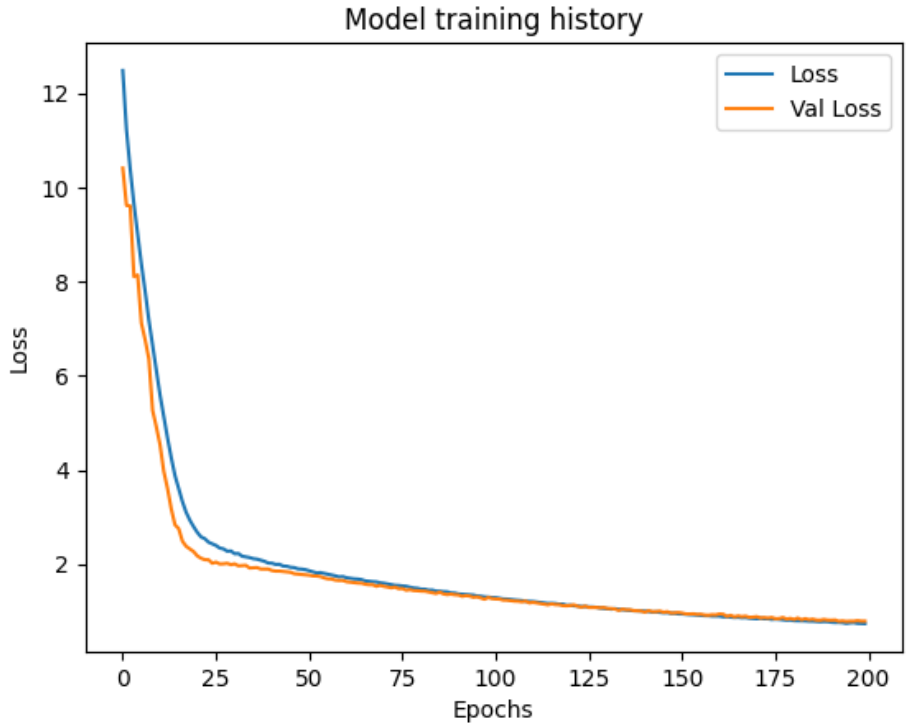

Figure 2. Training and validation loss curves over 200 epochs using the linear regression approach.

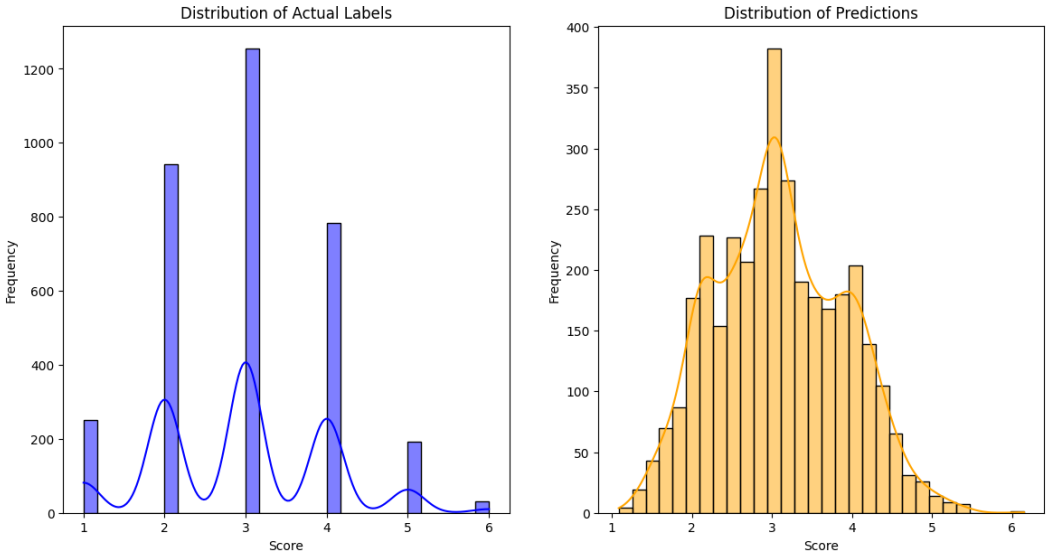

Figure 3. Distribution of Actual Labels and Predicted Scores using the linear regression approach

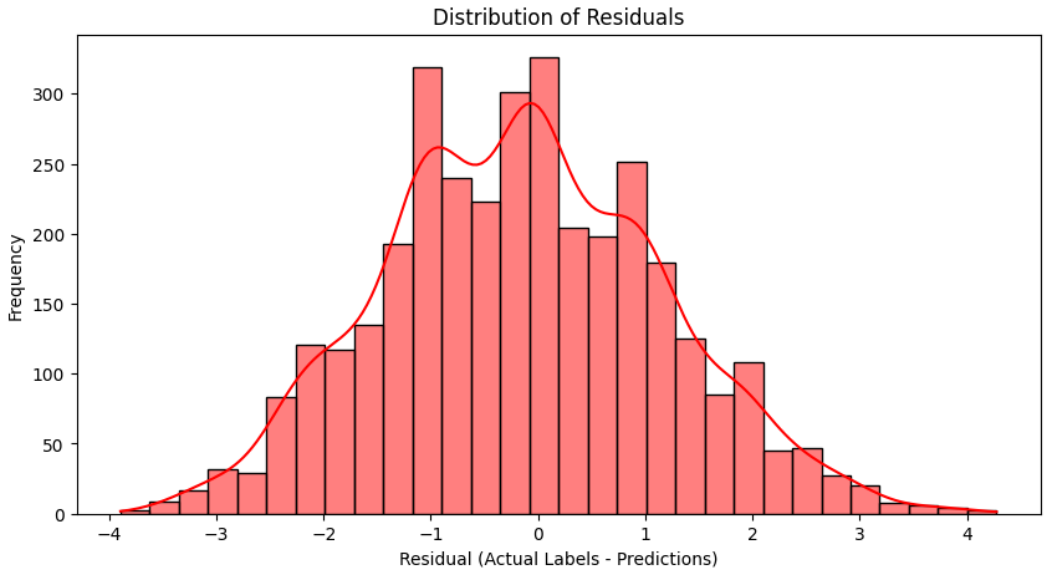

Figure 4. Distribution of the residuals using the linear regression approach.

Figure 5. Training and validation loss curves obtained fine-tuning the linear regression model to the IELTS dataset [9].

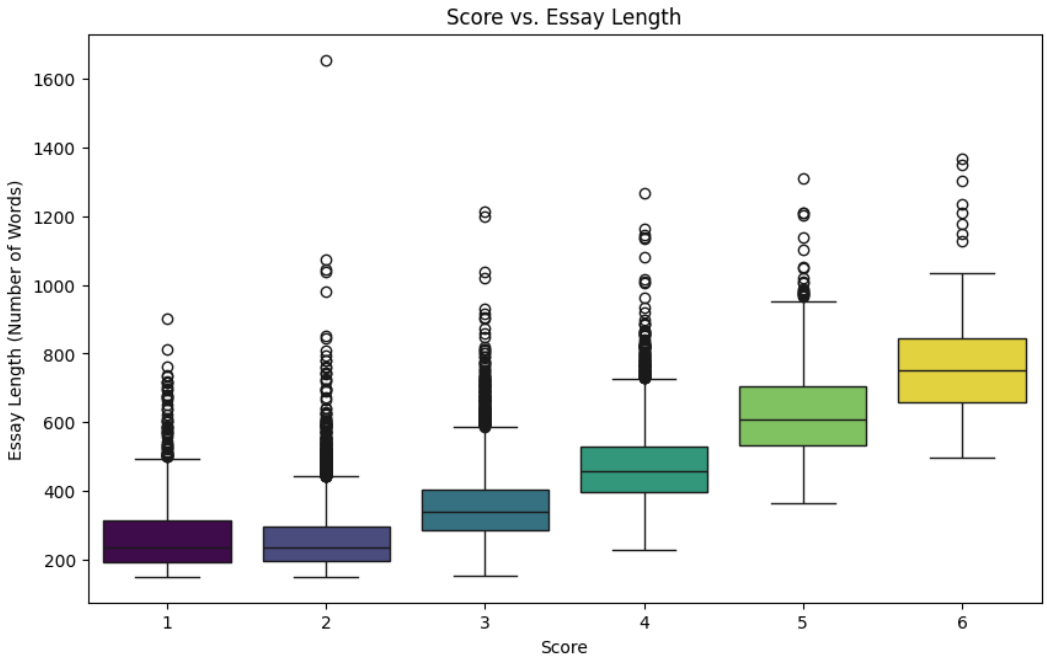

Figure 6. Box plot analysis of essay scores versus length from the Kaggle dataset [1].