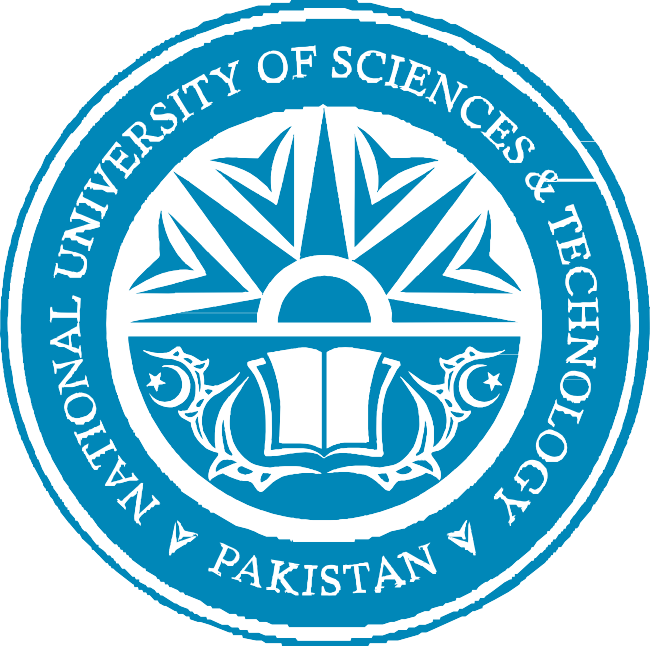
CS471 – Machine Learning

Fall 2024

Project Report



**Muhammad Abdullah - 393969**

**Azzan Rauf - 373199**

**Mohammad Saad Husnain - 365824**

**Syed Ahmad Shah - 369231**

# Project Report: Automatic Essay Scoring – Using NLP Libraries, ANNs and LSTM

## Submitted to: LE Munadi Sial

## Introduction

Automatic Essay Scoring (AES) is a process that utilizes machine learning (ML) and natural language processing (NLP) to evaluate and grade essays. The goal of this project was to design and implement an AES system using a dataset of 12,977 essays, leveraging deep learning models such as LSTMs and ANNs, combined with feature extraction techniques.

## Objectives

* Develop a model to automatically score essays based on a variety of linguistic and semantic features.
* Utilize advanced machine learning techniques such as LSTM and ANN for predictive modeling.
* Ensure adherence to constraints, including 30 extracted features derived from essays.

## Dataset

The dataset comprises 12,977 essays, with each essay labeled with a score representing its quality. Key attributes of the dataset include:

* **Essay text**: The raw text of the essay.
* **Final score**: The target variable for the models.

The dataset used was [The Hewlett Foundation: Automated Essay Scoring | Kaggle](https://www.kaggle.com/c/asap-aes/data).

### Preprocessing Steps

* Removal of stopwords using NLTK.
* Lemmatization using SpaCy.
* Sentence tokenization and word tokenization.
* Cleaning and normalization to reduce noise.
* Feature extraction to generate 30 key features.

Extracted features include features like:

* Average word length.
* Count of complex words.
* Readability indices (e.g., Flesch Reading Ease).
* POS (Part of Speech) tags distribution (Nouns, Adjectives, and Adverbs etc.)
* Sentiment analysis scores.
* Word count
* Sentence Count
* Spelling errors
* Readability using Flesh-Kincaid Score
* Punctuation
* Clause Density
* Semantic Coherence
* Transition word usage
* Use of Figurative Language
* Complex sentence ratio
* Question usage
* Argumentative Tone

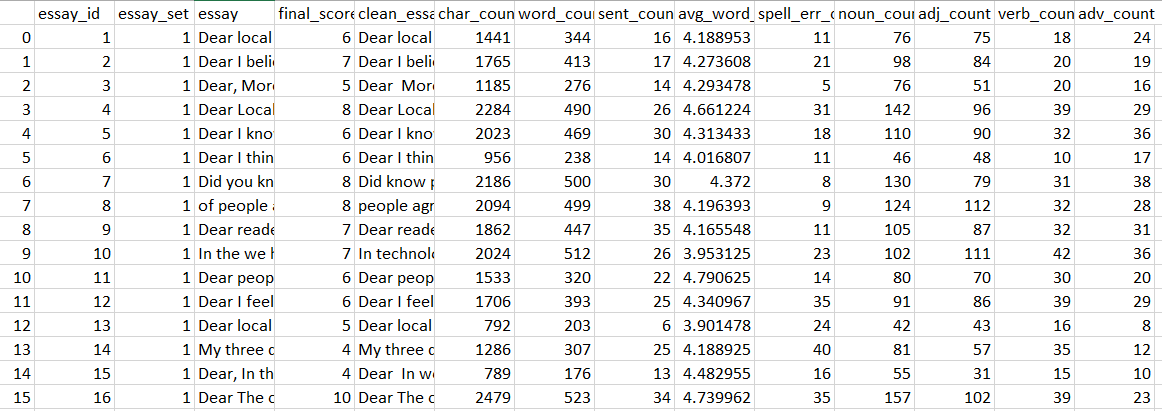


Figure . Dataset after basic feature extraction

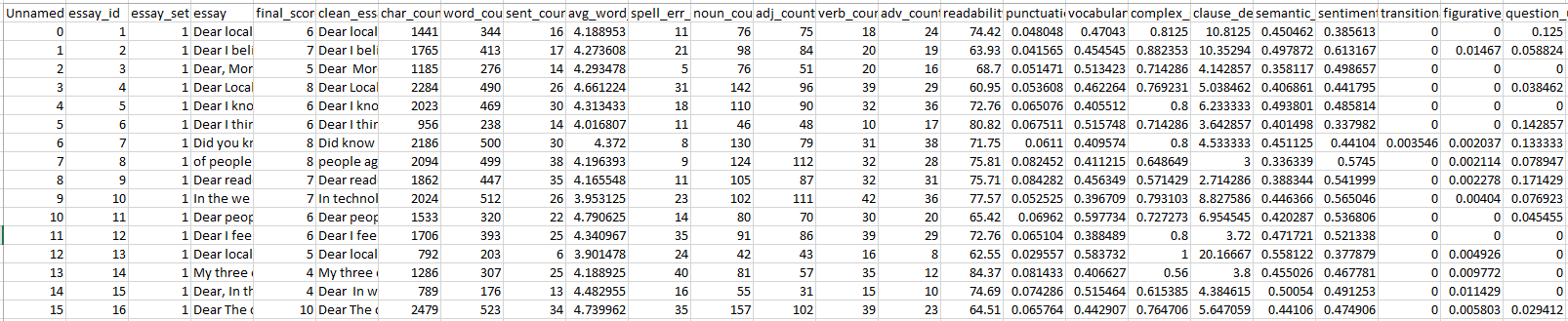


Figure . Dataset with complete feature extraction

## 4. Methodology

### Libraries Used:

* **pandas**: For handling tabular data, such as loading and manipulating the dataset.
* **nltk**: For natural language preprocessing tasks, including stopword removal and tokenization.
* **spacy**: For advanced NLP tasks like lemmatization and POS tagging.

### Data Cleaning:

* Removed special characters, redundant spaces, and non-alphanumeric content.
* Applied case normalization (converted all text to lowercase).

### Tokenization:

* Split essays into words and sentences using NLTK.

### Stopword Removal and Lemmatization:

* Removed common stopwords (e.g., "the", "is").
* Applied SpaCy's lemmatization to reduce words to their base forms.

### Features Extracted:

1. **Linguistic Features**: Word count, sentence count, and average word length .Complex word count: Words with more than three syllables.
2. **Readability Scores**: Flesch Reading Ease and Gunning Fog Index. Sentence length metrics.
3. **POS Tagging**: Frequency distribution of parts of speech (nouns, verbs, adjectives, adverbs).
4. **Semantic Features**: Sentiment analysis scores (polarity and subjectivity) using TextBlob.
5. **Custom Features**: Character counts and punctuation frequencies.

### Embedding Generation:

* **SpaCy's** pre-trained language model was used to generate embeddings for essays. These embeddings are high-dimensional vector representations of the text, capturing semantic information.
* Each essay was transformed into a **vector** of fixed size, enabling input to machine learning models.

### Data Preparation:

* **Feature Scaling**: Used StandardScaler to normalize feature values to ensure uniformity across features.
* **Train-Test Split**: Divided the dataset into training (80%) and testing (20%) subsets.

## 5. Model Training

### Libraries Used:

* **tensorflow** and **keras**: For building, training, and evaluating ANN, LSTM, and RNN models.
* **scikit-learn**: For baseline Linear Regression.

### Steps:

1. **Linear Regression**: Baseline model trained using the extracted features. Performance metrics (MSE, MAE, QWK) were calculated.
2. **Artificial Neural Network (ANN)**:

* Input Layer: Size matches the number of features.
* Hidden Layers: Two fully connected layers (64 and 32 nodes, ReLU activation).
* Output Layer: Single neuron for regression (linear activation).

1. **Long Short-Term Memory (LSTM)**:

* Input: Sequential embeddings reshaped for time-series analysis.
* Architecture: LSTM layer with 64 units. Dense layers for regression output.

1. **Recurrent Neural Network (RNN)**:

* Similar to LSTM but uses a SimpleRNN layer instead.

## 6. Evaluation

#### ****Libraries Used:****

* **scikit-learn**: For evaluation metrics like MSE, MAE, and QWK.

#### ****Steps****:

* Each model was evaluated using the test set.
* Performance metrics (MSE, MAE, QWK) were computed to compare models.









## 7. Alternative Models used

1. We also compiled an LSTM model and trained it purely on the essay text allowing it to automatically learn and extract features itself without explicitly making 30 features and training a model on them. This approach yielded better results but of course failed to meet the 30 feature constraint of the Semester Project Guidelines.



Figure . Evaluation metrics for alternative LSTM model

2. We also tried feature extraction from essays using BERT. We tried to use some of our extracted features along with multi-dimension embeddings provided by BERT, to fulfill the minimum 30 feature criteria. This multimodal approach did not yield very accurate results so we dropped this idea

## 8. Sample essay and score

Given sample:

AI is in education now. It personalizes learning and helps disabled students with tools like screen readers. Teachers don’t have to grade as much because AI does it. But there are problems like privacy and less human interaction. Still, AI will probably stay in education

