Use of Semantic, Syntactic and Sentiment Features to Automate Essay Evaluation

by

Harneet Kaur Janda
Lakehead University

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Lakehead University

Use of Semantic, Syntactic and Sentiment Features to Automate Essay Evaluation

by

Harneet Kaur Janda
Lakehead University

Supervisory Committee

Dr. Vijay Mago, Supervisor
(Department of Computer Science, Lakehead University, Canada)

Dr. Shan Du, Co-Supervisor
(Department of Computer Science, Lakehead University, Canada)

Dr. Yimin Yang, Departmental Member
(Department of Computer Science, Lakehead University, Canada)

Dr. Philippe J. Giabbanelli, External Member
(Department of Computer Science, Furman University, USA)

ABSTRACT

Manual grading of essays by humans is time-consuming and likely to be susceptible to inconsistencies and inaccuracies. Mostly performed within an academic institution, the task at hand is to grade hundreds of submitted essays and the major hurdle is the homogeneous assessment from the first till the last. It can take hours or sometimes even days to finish the assessment. Automating this tedious manual task is not only a relief to the teachers but also assures the students of consistent markings throughout. The challenge in automatizing is to recognize crucial aspects of natural language processing (NLP) which are vital for accurate automated essay evaluation.

NLP is a subset of the field of artificial intelligence which deals with making computers understand the language used by humans for expression and then further process it. Since essays are a written textual form of expression and idea exchange, automating the essay assessment process through a computer system leverages progress from NLP field and automates one of the biggest manual tasks of educational systems.

In recent years, an abundance of research has been done to automate essay evaluation processes, yet little has been done to take into consideration the syntax, semantic coherence and sentiments of the essay's text together. Our proposed system incorporates not just the rule-based grammar and surface level coherence check but also includes the semantic similarity of the sentences. We propose to use graph-based relationships within the essay's content and polarity of opinion expressions. Semantic similarity is determined between each statement of the essay to form these graph-based spatial relationships. Our algorithm uses 23 salient features with high predictive power, which is less than the current systems while considering every aspect to cover the dimensions that a human grader focuses on. Fewer features help us get rid of the redundancies of the data so that the predictions are based on more representative features and are robust to noisy data. The prediction of the scores is done with neural networks using the data released by the ASAP competition held by Kaggle. The resulting agreement between human grader's score and the system's prediction is measured using Quadratic Weighted Kappa (QWK). Our system produces a QWK of 0.793. Our results are repeatable and transparent, and every feature is very well explained as compared to other existing systems where authors have not explained the methodologies and feature extraction to a similar extent for the results to be

reproduced.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

First of all Dr. Vijay Mago, for providing me with the opportunity to work under his supervision and for supporting me through all the tough times. I was able to learn and grow as a researcher and also as an individual.

Dr. Shan Du, for the advice and guidance she has provided me throughout my time here at Lakehead University. I would also like to thank her for all the time she has devoted in helping me write my thesis.

Dr. Philippe J. Giabbanelli and Dr. Yimin Yang, for their patience and support in overcoming numerous obstacles in my thesis writing.

Datalab.science team, for helping me throughout my journey specially with proof-reading and revisions. I greatly appreciate all their help.

DEDICATION

I would like to dedicate this thesis to
my family and all my friends for always believing in me.

Chapter 1

Introduction

1.1 Overview

Essay writing is used in many academic disciplines as a form of evaluation. Multiple parameters are tested when evaluating these essays, ranging from grammatical mistakes to how meaningful the contents are [12, 62]. In 1966, Ellis Batten Page presented the idea of an automated system to grade essays based on his own experiences [77] and developed a system called Project Grade Essay® [78]. Although there have been plenty of innovations and advancements in the field since then, most of the existing systems aim to predict scores by taking into consideration extensive number of features which are mostly grammatical flaws, syntax errors and surface level semantic comparison using latent semantic analysis, tf-idf and content importance model [39, 49, 69]. In spite of using a large number of independent prediction variables extracted from the text, most of the systems fall short of in-depth analysis of semantic and sentiment analysis of the submission. In this thesis, we propose an Automated essay scoring (AES) system which uses a fewer number of high-quality, independent variables and provides the essence of the essay which is used to accurately predict the score. This thesis also proposes a graph-based approach to extract novel semantic features. We use semantic similarity algorithm [82] along with the vector embedding by FastText [80]. The semantic similarities between each statement are critical to scrutinize coherence and other semantic properties in the essay. We also employ the sentiment of the words used by the author of the essay while writing, which is vital for argumentative and persuasive essays.

Previous studies [24,99] have shown that when AES is compared with human graders about crucial characteristics of a good essay, the top responses are about the analysis of how the essay revolves around the question prompt, how well structured and sleek the information flow is, quality of grammar used, length, spellings, and punctuation. With respect to these responses, features are extracted from the essays and then the most influential ones are selected for an essay grading prediction model. The main strength of our proposed system is to use syntactic, semantic and sentimental features together in order to improve the accuracy of the already proposed systems while minimizing the number of features. Different prediction models are tested to find the one which works best for predicting the essay scores.

1.2 Motivation

Assessing and assigning a score to a large number of essay submissions which are written concerning an essay prompt is a tiring task. Many times, multiple raters are employed and trained which is expensive and extensive. Substituting the human raters with a computer system can save time, money and effort.

Human graders can be imperfect; they are susceptible to biases and irregularities based on other chores and activities they do in life [99]. Different human graders also have different grading styles and can also tend to give an overall higher grade just based on one good impression regarding a particular aspect. A computer system can overcome all these human shortcomings by uniform assessment throughout.

1.3 Problem Statement

Understanding human language is considered a laborious task due to its complexity. There are numerous ways to arrange words in a sentence. Also, words can have multiple meanings in different contexts. Therefore context-based knowledge is necessary to decipher the sentences correctly.
Making a computer understand the natural language used by humans is a stepping stone in Artificial intelligence field. With new cutting edge technologies and supervised learning models, there have been numerous works done in the area of AES to grade as accurately as human graders [99]. Our task is to develop a system which assesses and produces grades that solve the problems of manual grading and is based

on fair standards and gives us uniform assessment throughout. There is still room for automated systems to exceed the performance of human graders. The challenge is to find the right kind of features which form the basis of an intelligent system scrutinizing various aspect of natural language processing. Our system looks at the essay evaluation as a natural language processing task and makes sure to extract the syntax, semantic and sentimental information from the text and filter out the ones that are most informative.

1.4 Thesis Structure

This section provides a map of the dissertation to show the reader where and how it validates the claims made in the introduction chapter

Chapter 2 presents some background for the research presented in this thesis. This will help the readers to understand the work that has been done so far in the field. This chapter also gives information about the existing techniques and technologies that have been used by our system.

Chapter 3 talks about what is syntax analysis and its role in essay evaluation. The first part talks about the overview of syntactic features; the second part talks about methodologies used in the extraction of these features.

Chapter 4 contains a description of the semantic features used by the system. The first part talks about the details of these features; it further talks about methodologies used in the extraction. This chapter also talks about the algorithm used in computing semantic similarities.

Chapter 5 contains information about implementation and evaluation. This chapter also throws light on the essay data-set. This chapter also talks about the selection of important features and techniques used in finding the important variables. Results from different learning models are shown, and the final comparisons with other existing systems are shown.

Chapter 6 talks about the conclusion we derive from our research. We also briefly mention the future works that could help improve the system.

Chapter 2

Background

2.1 Brief Description of Methodologies

For efficient processing of text, different areas of natural language processing (NLP) domain, i.e., syntax, semantics, and sentiments are analyzed by the existing systems [99], Figure 2.1. In terms of meeting the goal of automated essay evaluation, systems extract features, and further use them with machine learning to predict the scores. Following is a brief description of methodologies used in the literature with essay evaluation systems:

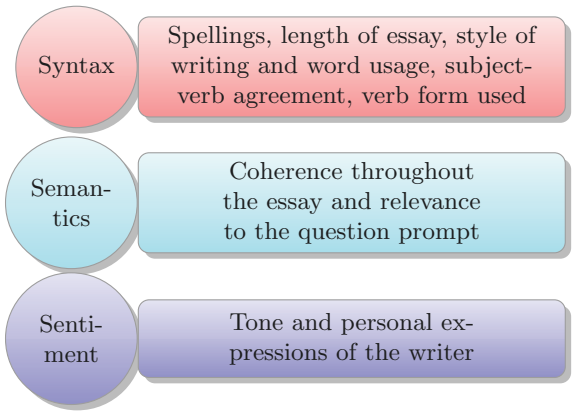


Figure 2.1: Overview of syntax, semantics and sentiments involved in an essay

2.1.1 Machine Learning

Machine learning enables systems to learn from their experience. It is one of the biggest revolutions in computer science, and research has benefited from a machine learning program which optimizes its results while learning during execution [67]. Figure 2.2 shows machine learning can be categorized into `supervised` and `unsupervised`

learning depending on whether or not the program learns from already labeled data. Each category has several algorithms that can be used to train the machine depending on the kind of problem statement and results expected from it. Machine learning can be further divided into classification, regression, and clustering. The three mentioned machine learning techniques further comprise of many algorithms.

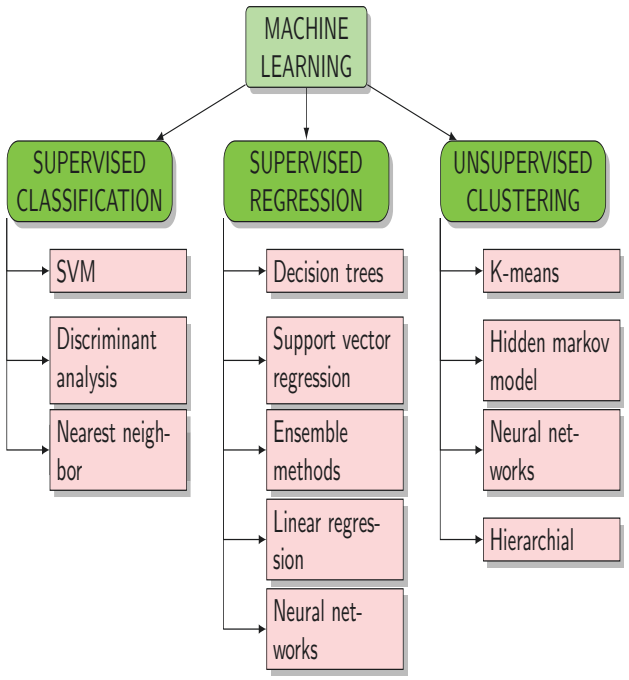


Figure 2.2: Overview of selected machine learning techniques

Supervised Learning

Supervised learning deals with the branch of machine learning where algorithms learn from labeled training data, analogous to the process of learning under a supervisor [71]. A supervised learning algorithm uses the independent input variables (X) and its corresponding dependent output variable (Y) to train a mapping function. The purpose of training the algorithm is to make it capable enough to make predictions on unseen situations. Supervised learning algorithms can be further divided into two categories: `classification` and `regression.`

`Classification` refers to predicting labels or categories. These algorithms can be multi-class or binary. These algorithms generate a continuous value output repre-

senting the probability of a given class, and this value represents the confidence level of the decision, which is further used to make final predictions. `Regression`, on the other hand, uses a set of inputs to generate a continuous variable as output, which is a real value [60].

The propose of our system is to give grades (i.e., real numbers) to an essay; this purpose can be treated as a `Regression` problem as we are trying to predict a continuous output. Figure 2.3 shows the framework for the process of supervised regression technique.

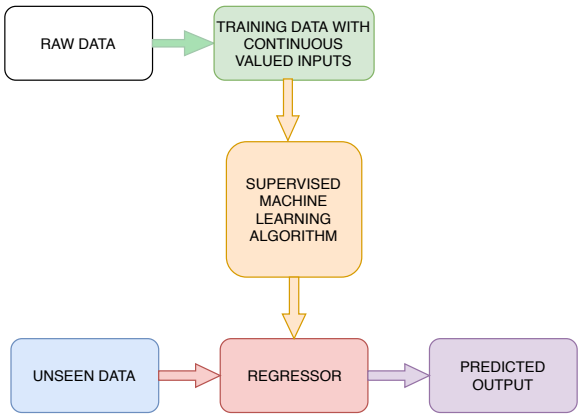


Figure 2.3: Supervised regression technique

2.1.2 Syntactic Analysis

The earliest developments in the AES field were focused on the correctness of syntax of essays including grammar, spellings, length, etc. [99]. Syntax mainly refers to the arrangements in which words are put together. During the assessment of essays, higher scores are usually given to more complex and deeper meanings. However, making a computer understand this deeper meaning as a human would do is challenging. The way a sentence is formed is called its the syntax. Different people might express the same things differently. Do you make the sentence long with many dependent clauses? Do you make your sentences short? Do you repeat the same word again and again? Answers to these and other syntax related questions make every submission different. We need to extract syntactic information from essays in the form of independent variables to be fed to the machine learning algorithm. Grammatical rules are protocols which we need to consider to define the structure as well as the correctness of the sentences. Grammatical rules are applied together to a group of words and not just an individual word in a given sentence [86]. This analysis is also commonly known as parsing or simply syntax analysis. This analysis involves fragmenting a text into its `part of speech` and establishing a relationship with each part [15].

2.1.3 Semantic Analysis

In natural language, semantic analysis is about understanding the meaning of what is written in a particular text [32]. The semantic part of language processing tries to understand if the formation of the sentences, occurrences of the words in a written/oral communication make any sense or not. Semantic similarities are useful to understand the coherence of the essays and their relevance to the question [112]. Recent works in the NLP area to provide solutions for calculating semantic similarities can be categorized as following methods :

- Word co-occurrence methods [65] [25].
- Similarity based on a lexical database [56].
- Method based on web search engine results [7].
- Methods based on word vectors using recursive neural networks [37, 73, 109].
- Unsupervised approaches [38, 83].

Unsupervised approaches to determine semantic similarities require less computational resources. The unsupervised method used by Pawar et al. mentioned above outperforms other methodologies as per the Rubenstein and Goodenough (R&G) benchmark standard [96] [83]. This algorithm uses an edge-based approach using a lexical database and incorporates corpora-based statistics into a standardized semantic similarity algorithm. It is divided into two modules of maximizing the similarity and bounding the similarity.

Maximizing the similarity This is the first pass of the algorithm. It can be further divided into three categories:

- Word Similarity: To compute word similarity the proposed algorithm uses *WordNet* [68]. *WordNet* has path relationships between noun-noun and verb-verb only. Every word in *WordNet* has *synsets* according to the meaning of the word in the context of a statement.
- Sentence Similarity: Semantic value vectors for the sentences are formed using the best matching results from all the *synsets*. The size of the vector is determined, based on the number of tokens/words which are nouns and verbs. Let's

see an example:
Sentence 1 = "In this modernized world computers are key."
Sentence 2= "These benefits have positive effects on people throughout the world and are why I support the advances in technology."
The formed semantic vectors contain semantic information concerning all the words from both the sentences. For example, the semantic vector for Sentence-1 is:
V1 [0. , 0.90926098, 0.15404895, 0.06554339, 0., 0., 0., 0., 0.]
Similarly, V2 has semantic information of sentence-1 and sentence-2.
V2 [0.12583497, 0.07004909, 0.15633548, 0.39054567, 0.15404895, 0.90926098, 0.13802896, 0.13802896, 0.12132365]
To compute the semantic value S, the magnitude of the normalized vectors is used, Equation 2.1.

$$S = ||V1||.||V2|| \tag{2.1}$$

- Word order similarity: If the same words are used in two sentences in a different order the word order similarity is taken into consideration.

Bounding the similarity The first pass of the algorithm maximizes the similarity. The second part bounds the similarity by taking into consideration the syntax of the sentence like negation. Approaches used to bound the similarities are :

- Recurrence of words.
- Negation of Words.
- Spacy's dependency parser model

The final semantic similarity value derived for the two sentences given above is 0.3155.

2.1.4 Graph-based Methods for Text Analysis

Author Pillutla, Venkata Sai Sriram used graph-based representation to depict the associations between sentences in a text using cosine similarity [87]. Jin et al. used graph-based representation in text mining task to detect unapparent links between concepts across documents [46]. Graph edges representing order-relationship between

the words represented by nodes have been used for text summarization [58]. Giabbanelli et al. used analysis of casual maps to assess problem solving skills of students [30]. Gupta et al. assess student learning in form of causal networks or maps [35]. Giabbanelli et al. use the assessment of knowledge maps to study student's knowledge for an ill structured problem [31].

2.1.5 Sentiment Analysis

Sentiment analysis also is known as opinion mining, is a process to find the emotional tone which can be very useful to understand the attitude of the writer towards the subject. Typically, sentiment analysis systems have been applied for use with reviews and social media analysis. Typically, there are two main research pathways within sentiment analysis. First is the lexical approach that focuses on building lexical dictionaries and the second is machine learning [111]. There has been only a little done towards building a sentiment analysis system to identify sentiment and polarity in student or test-taker essay writing [44].

Sentiment Analysis is the study of opinions, attitudes, and emotions toward an entity [72]. Each writer has a unique or changing tone towards the subject being written about [98]. For example, if a writer wants to write about his disagreement about a scenario, this analysis tells us about how negative the language used is or how positive or neutral the tone of the writer is. Sentimental analysis plays an important role in AES. The factors making it important are listed below:
1) In argumentative and persuasive essays as an author needs to defend and prove his/her point of view on the subject, their tone and the way of textual sentiment expression affect their writing [13].
2) The sentiment analysis of figurative speech in the essays helps us know the polarity inclination they contribute to the text which is otherwise difficult to comprehend by the computer [1, 90].
A study on existing sentiment analysis methods [92] show that VADER is one of the best performing open-source sentiment analysis tool. We will elaborate on VADER's functionality in the sub-section below.

VADER

Valence Aware Dictionary and sentiment Reasoner (VADER), which is a rule-based model for sentiment analysis [43]. It is fully open-sourced under the MIT License. It does not require any training data but is constructed from a valence-based, human-curated gold standard sentiment lexicon [93]. It uses a lexicon of words already trained as per their sentimental inclination as positive, negative or neutral. VADER uses Amazon's Mechanical Turk to get scores for the lexicon [11]. The polarity of each sentence is checked, and final polarity values towards being positive, negative and neutral are obtained in terms of percentage. The positive, negative and neutral scores represent the chunk of text that falls in these categories. This means if our essay was rated as 50 percent Positive, 25 percent Neutral and 25 percent Negative, these values should add to a 100 percent.

Following are the important key points used by VADER to analyze sentiments, refer

Sentence	**Values of Sentiment based features**
This car is good	Negative : 0, Positive: 0.492, Neutral : 0.508
This car is good!	Negative : 0, Positive: 0.433, Neutral : 0.567
This car is good!!!	Negative : 0, Positive: 0.486, Neutral : 0.514
This car is GOOD!	Negative : 0, Positive: 0.472, Neutral : 0.528
This car is extremely good	Negative : 0, Positive: 0.492, Neutral : 0.508
This car is marginally good	Negative : 0, Positive: 0.442, Neutral : 0.458
This car is good, but it's fuel economy is bad	Negative : 0.31, Positive: 0.192, Neutral : 0.558
This car is not good	Negative : 0.376, Positive: 0, Neutral : 0.624

Table 2.1: Polarity values of same sentence under different contexts

Table 2.1 for the polarity values given by VADER to the sentences:

- Punctuation: The use of punctuation like exclamation marks, enhances the intensity without modifying the semantic orientation of the sentence. For example, *This car is good!* is more intense than *This car is good.* The number of such punctuation used is directly proportional to the magnitude of expression.

- Using upper case: Using upper case letters to emphasize or stress on a sentimental word in the presence of other lower case letters, increases the magnitude of the sentiment intensity. This is not an acceptable approach in essay writing but occurs at times in high school writing.

- Degree modifiers: Use of degree modifiers or intensifiers impact the polarity of expression. For example, *This car is extremely good.* is more intense than *This car is good.* whereas *This car is marginally good.* reduces the intensity.
- Conjunctions: Use of conjunctions like *but* can shift sentiment polarity. For example, *This car is good, but its fuel economy is bad* has mixed sentiment, with the second part of the sentence being more polar than the first.
- Preceding Tri-gram: Examining the tri-gram preceding a word is important for checking any negations. Usually, negations flip the polarity values of a certain extent.

As shown in Table 2.1, three sentiment based features can be extracted.

2.2 Related Works

The history of automated writing evaluation goes long back. In January 1966, Ellis Batten published an article emphasizing on the possibility and importance of an automated essay evaluation system [77]. Two years later, he successfully developed Project Grade Essay ® [78] which uses statistical methods and multiple linear regression. Intelligent Essay Assessor (IEA) developed by Peter Foltz and Thomas Landauer in 1997 uses natural language processing (NLP), latent semantic analysis (LSA) and Machine learning for the prediction [64]. Latent semantic analysis [54] refers to extracting context-based word meaning from a large corpus by statistical methods. In 1998, a system called Intellimetric ® [97] was developed which uses NLP methods and mathematical models for predicting grades. The educational and testing services use e-rater ® [12] for generating scores and feedback which uses NLP to produce features which are combined with a statistical model to predict the scores [12]. Bayesian Essay Test Scoring system developed by Lawrence Rudner uses Bayesian network and statistical methods. The system CRASE® [62] also uses NLP and ML.

2.2.1 Semantic Analysis in Essay Evaluation

Semantic information on written text tells about coherence and closeness of the information flow within the essay and with the question prompt. There have been

attempts to extract semantic information using various NLP techniques in AES. Existing systems measure coherence in the text using different supervised and unsupervised approaches. Usually, the unsupervised approaches measure lexical cohesion, i.e., repetition of words and phrases in an essay. Foltz et al. [27] assume that coherent texts contain a high number of semantically similar words and measure coherence as a function of relatedness between adjacent sentences. Some systems have used latent semantic analysis (LSA) [27, 69], probabilistic latent semantic analysis (PLSA), and latent Dirichlet allocation (LDA) [40]. LSA, PLSA, and LDA are topic modeling techniques, i.e., class of text analysis methods that analyze groups of words together to capture how the meaning of words depends on the broader context in which they are used in natural language. Systems have also used unsupervised approaches like usage of similar words and sentences in an essay to depict the level of coherence [28]. Klebanov et al. [50] aimed to predict the score for an essay based on its relatedness to Content Importance models. Higgins et al. [39] measured coherence features and incoherence through semantic similarity between essay question and discourse elements of the essay. Zupanc et al. [112] incorporated coherence features to convert parts of essay into a semantic space and measure various characteristics, but the authors failed to provide enough information for other researchers to repeat their results. Semantic relationship between chunks of text can be represented as a graph. The Graph-based features can help obtain information about the coherence in the text by pattern recognition [17].

2.2.2 Sentiment Analysis in Essay Evaluation

Distinctive opinions and polarity of words used by the writer in an essay shape up the overall essay construction and quality, specifically in persuasive and argumentative essays [59]. Many NLP tasks have used sentiment analysis such as in social media [8], movie's reviews [79], news and politics [94, 102]. One of the first attempts at incorporating sentiments in AES involved using subjective lexicon(s) to get the polarity of the sentences [5]. Some other noteworthy works are finding argumentative discourse in persuasive essays [104] where authors proposed to classify argument components as support or not to obtain argumentative discourse structures. Another prominent work [4] found the sentiment of sentences in essays by examining the sentiment of multi-word expressions. Farra et al. [26] matched up to the opinion expressions in the essays to their respective targets and use features extracted to predict the scores.

Chapter 3

Syntactical Attributes

3.1 Methodologies used in Extraction of Features

Syntax refers to the order/arrangement of content. Word classes, largely corresponding to traditional parts of speech (e.g, noun, verb, preposition, etc.), are syntactic categories. In phrase structure grammars, the phrasal categories (e.g, noun phrase, verb phrase, prepositional phrase, etc.) are also syntactic categories. In this part, we mainly use Natural language processing toolkit (NLTK) [1] [61], which is a python based platform to extract language-based data. NLTK also provides functions like classification, tokenization, stemming, tagging, parsing which are very helpful to extract the syntactic features of a text.

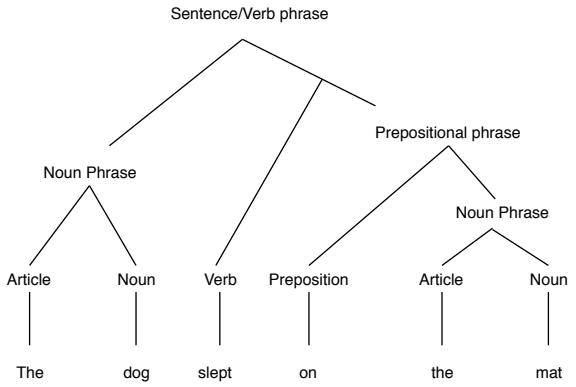


Figure 3.1: Constituent structure of a sentence displaying each part of speech

We take into account subject-verb agreements, repetitions, and word length and many more syntactic features. To analyze the syntax, a submitted essay is segmented into sentences by given full stops. These sentences are further tokenized into words to analyze each word the essay is composed of, Figure 3.2. To obtain a set of syntactic features, we selected the ones which are widely used by researchers in the literature

[1] https://www.nltk.org/

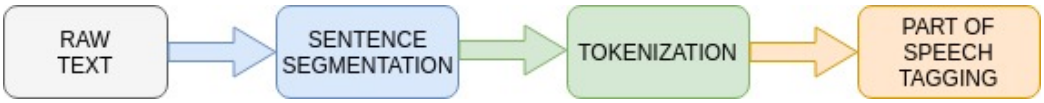


Figure 3.2: Segmentation and tokenization of the essay to determine part of speech used

[14, 21, 63]. To understand the syntactic structure of an essay, we count the total occurrences of the following syntax related features:

1) **Unique part-of-speech used**: Over repetition of words of the same `part of speech` in an essay is regarded as an inefficient use of English grammar. It is a common mistake made by non-native speakers in their writing [91]. For example, *Tom is a student. Tom is a good guy.* indicates an excessive use of nouns, where a pronoun could have been used instead. Figure 3.1 shows a sentence after `part of speech` tagging is done. We can see there are four unique `part of speech` used in the sentence. In the Listing 3.1, an essay is tokenized into words using *word_tokenize* function from *NLTK* and each word is tagged to a part of speech using the function *pos_tag*. The *found* tags are put into sets. A set is collection with no duplicates. Length of the set is obtained which represents the total number of unique `part of speech` in the essay. We believe using a very limited `part of speech` in an essay can lead to less score.

Listing 3.1: Code snippet to count the part of speech from a essay

```
def syntatic(essay):
    sentences = essay.split('.') #Split essays
                                    by given full stops
     jjs, pdt, cc, ex, jjr = 0
     for s in sentences:
      for word,pos in nltk.pos_tag(nltk.word-tokenize(s)):
          if(pos == 'JJS'):
           s_adj += 1  #Using word tokenize
                          and pos-tagging by nltk
           #increment counter of superlative adjectives
          if(pos == 'PDT'):
           pdt += 1
           #increment counter of predeterminers
```

```
        if(pos == 'CC'):
         cc += 1
         #increment counter of coordinating conjunctions
        if(pos == 'EX'):
         ex += 1
         #increment counter of existential there
    return jjs, pdt, cc, ex
```

2) **Misspelled words**: Use of wrong spellings can lead to misinterpretation of the word by the essay evaluator. Also, this is one of the most common mistakes naive writers make [23]. A dictionary for American English called *en-US* is used upon the spell check library pyEnchant that works on Levenstein Distance algorithm [3] to find the number of misspelled words per essay. In the Listing 3.2, we use the *check* function from spell checker library and count the total occurrences of misspelled words.

3) **Existential** *there*: Existential *there* is used to indicate the presence of existence of an entity [2]. Huckin et al. [42] stated that expert writers use the word *there* for only important linguistic purposes like to emphasis existence, to state new information, topics, and to summarize. Therefore, we believe excessive use of existential there can lead to low score and should be avoided. In the code Listing 3.1, we count the number of existential there using *pos_tag* function from *NLTK* the acronym for a existential there is *EX*. When a `part of speech` tag is found to be *EX*, the counter for existential there is incremented.

4) **Superlative adjectives**: These are used when a subject is compared to three or more objects and usually have a suffix -est added to it. For example, *sweetest* and *brightest*. In the code Listing 3.1, we count the total occurrences of superlative adjectives. The tag used for superlative adjective is *JJS* by the function *pos_tag*. When a `part of speech` tag is found to be *JJS*, the counter for the superlative adjective is incremented. We believe the use of superlative adjectives is a good practice over the use of intensifiers like very really, fairly etc.

5) **Predeterminers**: Predeterminers are used before determiners or article that gives more information about the noun in the sentence. For example, in the sentences *all these students* and *once a week* the words *all* and *once* are predeterminers. Taguchi et al. have highlighted the importance of linguistic features as an indicator of writing quality and predeterminers play an important role in it [108]. Refer to Listing 3.1, the acronym for a predeterminer is *PDT*. When a `part of speech` tag is found to be *PDT*, the counter for the superlative adjective is incremented.

6) **Coordinating conjunctions**: Coordinating conjunctions are used to join two main clauses. For example, 'My dog Tom has beautiful eyes *but* a notorious personality'. Here *but* is the coordinating conjunction joining two main clauses. The higher frequency of coordinating conjunction used to link sentences plays a significant role in the overall length of paper [29]. As coordinating conjunctions make the sentences larger, we believe larger sentences become harder to understand and leads to low scores. We count the total number of coordinating conjunctions used in the essay. The code Listing 3.1, the acronym used for coordinating conjunction is *CC*. When a `part of speech` tag is found to be *CC* for a word by function *pos_ tag*, the counter for coordinating conjunction is incremented.

7) **Words ending with -ing**: Nouns and verbs ending with -ing are known as gerunds and participles respectively. Excess use of them makes the writing look naive. Even though using these words is not grammatically wrong, it is advised to choose your word suffix wisely and re-using -ing throughout is condemned [106]. We count the total number of words ending with ing using regular expressions [107]. We use the library called *re* [2] in python . In the code Listing 3.2, we use regular expression '\b(\w+ing)\b' to identify words ending with -ing and count the occurrences in each sentence and add them to the total counter.

[2]https://docs.python.org/3/library/re.html

8) **Common sentence length**: Each sentence needs to be of an average length. A sentence too long or too short reflects poor writing and makes the comprehension difficult for the readers [74]. There is an inverse relationship between the grade awarded and difficulty to understand [16]. We believe very long or very short sentences are harder to understand and lead to low scores.

9) **Subject-verb agreement**: Using a singular subject with a plural verb or vice-versa leads to subject-verb disagreement. The total number of singular subjects, plural subjects, and number of different verbs forms are counted. Making these errors are mostly common with non-native english speakers [105].

10) **Repetitive words**: While there are some words that be can safely repeated in a sentence, repeating the same word again and again can distract the grader from the point that writing is trying to make [91]. Being cautious with repetition makes writing more professional. We count the maximum number of repetitions that occurred for a word in an essay. In Listing 3.2, we use the function called *allWorddist* and count the occurrences of most common words using function called *most-common*.

Listing 3.2: Code for extracting syntactic attributes from essay

```
def frequency(essay):

    list = nltk.pos_tag(word_tokenize(essay))
    # using word tokenize by NTLK
    unique_pos = len(set([x[1] for x in list]))
    # Set is a collection with no duplicates
    wordslist = essay.split(' ')
    # Split essay by spaces
    words = len(wordslist)
    # Counts total number of words
    characters = len(essay)
    # Counts total number of characters
    common = [len(i[0]) for i in
```

```
            allWordDist.most_common(1)]
        # allWordDist counts occurences of every word
            and finds most common using NLTK
sentences = text.split('.')
# split by full-stop
count_ing = 0
for sentence in sentences:
    list_ing =re.findall(r' \b(\w+ing)\b', sentence)
        # Putting words ending with -ing into
            a list using regular expression
      count_ing += len(list_ing)

for words in wordlist:
 if check(words) == false:
        misspelled += 1
        # Using Spell checker library

   return unique_pos, words, characters,
          common, count_ing, misspelled words
```

11) **Words**: Too short or a too long essay can reflect on the scores. Counting the total number of words in an essay plays a crucial role on scoring. We count the total number of words used in the essay. In the Listing 3.2, we put all the words from the essay into a list and count the length of the essay's word list using the python function *len*. As per the common norm, essay writing tests have a word limit associated where students are asked to not write more or less than certain number of words. We believe using less or more than the word limit affects the scores.

12) **Characters**: Apart from counting the total number of words, it is also important to keep a check on character count as it highlights the total use of alphabets

as well as the non-alphabet elements. Total *character count* includes the count of alphabets, punctuation, numbers, and spaces. We believe if a student uses appropriate number of words but extra spaces or unnecessary punctuation it increases the character count which is not a good practice. In many essay writing contests, a limit is set on the characters to be used and this feature impacts the score. We count the total number of characters used in the essay using the python function *len*, code Listing 3.2.

Chapter 4

Semantic Attributes

4.1 Overview

Semantics is also known as the study of meaning. The main purpose of any language is to communicate meaning to one another [32]. Semantic attributes are an important aspect of NLP that contributes to its meaning. Semantic analysis is a method for extracting and representing the contextual meaning of words or a set of words. Even a very well structured and grammatically correct essay also needs to be well coherent to qualify for good grades. To check the semantics associated with it, we need to make sure the content of the essay does justice to what the question prompt says, and the content flow in the essay's sentences are meaningfully related.

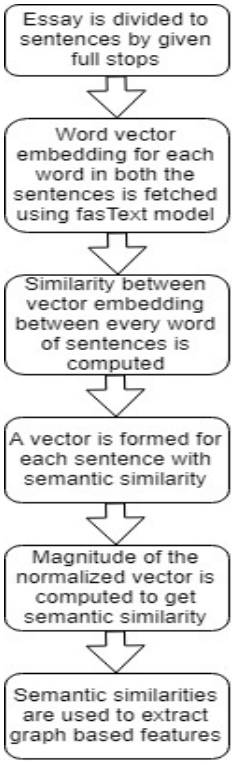


Figure 4.1: A top-down chart describing steps of computing sentence similarities and semantic properties of an essay.

It is crucial that every piece of an essay fits together semantically. Incoherence of a part of the essay with other sub-parts indicates that the particular part is unconnected to rest of the essay [70]. It is an important criteria during essay evaluation that essay is organized around a central unifying theme [74]. To check the coherence, we compute semantic similarities between the sentences of an essay. We believe comparisons only between consecutive pairs is not enough and we decided to make a semantic comparison between every sentence in the essay. In our research, we propose to compute semantic similarity between not just consecutive statements but also between each pair of sentences to analyze the over all coherence of the essay 4.2. The similarities between all the sentences help us derive novel graph-based features which highlight how different essay sentences are connected semantically and their patterns .

4.1.1 Computing Semantic Similarities

Word embeddings are mapping of phrases or words in a sentence to vectors [20]. To get the word-vector embeddings, we use embeddings by fastText created by Facebook's AI research lab [48] with *Magnitude* which is a vector utility library [80] and a faster alternative to Gensim [89]. The files with *.magnitude* extension for vector embeddings are designed to allow lazy loading that supports faster look-ups. Lazy loading refers to deferring of initialization of an object until the point at which it is needed. We compute the sentence similarities by providing these vector embeddings to the semantic similarity algorithm [10, 81–83], Figure 4.1 and Figure 4.3. The proposed solution by Pawar et al. as explained in the Chapter 2 uses *Wordnet* which considers only noun-noun and verb-verb connections. To overcome this and make the algorithm suitable for each `part of speech`, our algorithm considers the similarity between every word of the first given sentence with every word of the second sentence

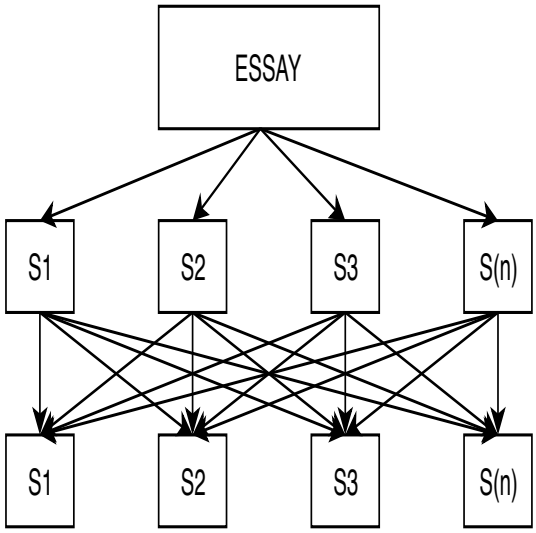


Figure 4.2: Semantic comparisons between each sentence of the essay

using the vector embedding values. The semantic vectors *V1* and *V2* contain semantic information concerning the words from both the sentences. These two vectors are normalized using their magnitude. The final similarity value is obtained after considering recurrence of words, negation, and Spacy's dependency parser model. The algorithm is explained in Chapter 2- Background.

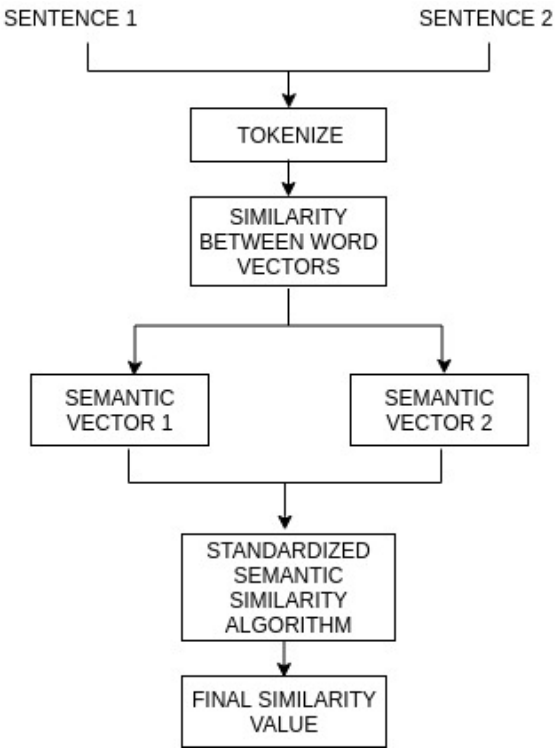


Figure 4.3: Using word vector embedding by fastText with standardized semantic similarity algorithm

When an essay is given as an input to the system, it is split into sentences by given full stops. We perform pre-processing and convert it to lower-case and remove the stop words. Each sentence from the essay is compared to every other sentence, Figure 4.2. Each pair of sentences in the essay are tokenized, and similarity between their word vector embedding are computed using *similarity* function by the *Magnitude* library in python, Algorithm 1. A vector is formed for each sentence using *word2vec* function, and these semantic vectors are used by the semantic similarity algorithm, Figure 4.3. The result of this algorithm is between 0 and 1.

Algorithm 1: Incorporating Word2Vector embeddings in the semantic similarity algorithm

```
function word2vec(word-1, word-2);
Input  : Two sentences s1, s2 from the essay
Output: semantic vector for sentence 1 and sentence 2
vectors ← Magnitude(word_vector_embedding.magnitude)
for word w1 in s1: do
    ; // Iterating through words in both sentences
    max = 0
    temp_list = []
    for word w2 in s2: do
        word_similarity = vectors.similarity(word-1, word-2); // using
         similarity function from Magnitude library in python
        temp_list.append(word_similarity)
        max = max(temp_list)
    end
    semantic_vector.append(max)
end
return semantic vector for sentence 1 and sentence 2
```

EXAMPLE ESSAY

Dear Newspaper, I think that the effects are okay as long as we get off the computers and go outside and see some friends and get some exercise. Computers let us not just talk to each other but it also lets us challenge each other on games without hurting each other it could even stop all ways all at once because we could challenge other countries in war games without killing real living people.We can look up how to stop snake venom from getting to your heart and how to make a how and some arrows to hurt with.It also makes it easier to find health insurance, car insurance, and house insurance. We can check our taxes and stocks.We can look up historical facts on the computer. You can find plumbers, technicians, oil companies, and lumber companies. you can find dates on the computer, too and find information about certain eople too.

This essay is comprised of 8 sentences. Following are the sentences split by given full-stops:

a. Dear Newspaper, I think that the effects are okay as long as we get off the computers and go outside and see some friends and get some exercise.

b. Computers let us not just talk to each other but it also lets us challenge each other on games without hurting each other it could even stop all ways all at once because we could challenge other countries in war games without killing real living people.

c. We can look up how to stop snake venom from getting to your heart and how to make a how and some arrows to hurt with.

d. It also makes it easier to find health insurance, car insurance, and house insurance.

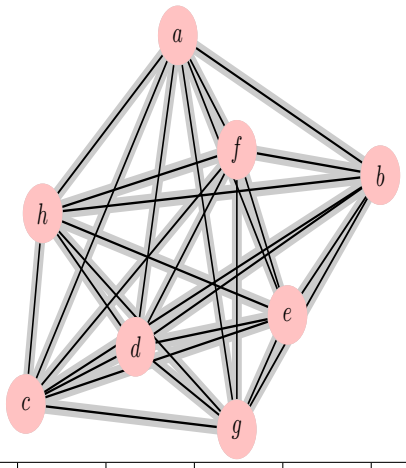
e. We can check our taxes and stocks.

f. We can look up historical facts on the computer.

g. You can find plumbers, technicians, oil companies, and lumber companies.

h. you can find dates on the computer, too and find information about certain eople too.

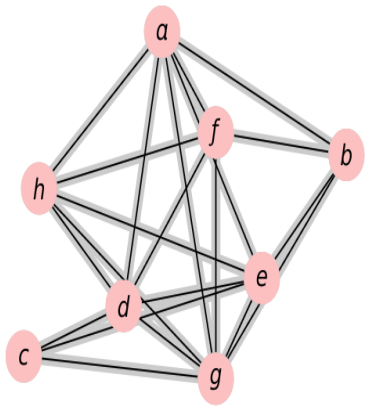
Semantic similarity computed between each of these sentences can be seen in the Table and Figure 4.1. Each of the semantic similarity value ranges from 0 to 1.



Sentences	a	b	c	d	e	f	g	h
a	1.0	0.56	0.19	0.59	0.56	0.76	0.56	0.60
b	0.56	1.0	0.34	0.35	0.46	0.55	0.50	0.25
c	0.19	0.34	1.0	0.71	0.52	0.28	0.64	0.32
d	0.59	0.35	0.71	1.0	0.49	0.60	0.69	0.79
e	0.56	0.46	0.52	0.49	1.0	0.57	0.72	0.66
f	0.76	0.55	0.28	0.60	0.57	1.0	0.57	0.61
g	0.56	0.50	0.64	0.69	0.72	0.57	1.0	0.68
h	0.60	0.25	0.32	0.79	0.66	0.61	0.68	1.0

Figure 4.4 & Table 4.1: Graph based representation of sentences using semantic similarities between sentences as edge weight and table of semantic similarity values between each sentence

Each sentence is compared to every other sentence in the essay to check the semantic similarity between them (a value between 0 and 1). Several researchers in the NLP field have used sentences or words transformed to graphs for text summarizing [87, 101]. Deriving motivation from such works, we propose a novel approach to represent semantic similarities between essay sentences as graphs and deriving features from these graphs. Considering each sentence as a vertex and the similarity values as edge weights, the results obtained are transformed into a fully connected graph to view the relations on a spatial space, Figure 4.4 and Table 4.1. For some graph-based features, it is important to use only the strong semantic relations as weak connections can affect the features to a large extent and thus overshadow other powerful connections. To make sure that only the relevant and meaningful connections are considered, every connection with a similarity value less than a certain threshold is dropped to obtain



Sentences	a	b	c	d	e	f	g	h
a	1.0	0.56	-	0.59	0.56	0.76	0.56	0.60
b	0.56	1.0	-	-	0.46	0.55	0.50	-
c	-	-	1.0	0.71	0.52	-	0.64	-
d	0.59	-	0.71	1.0	0.49	0.60	0.69	0.79
e	0.56	0.46	0.52	0.49	1.0	0.57	0.72	0.66
f	0.76	0.55	-	0.60	0.57	1.0	0.57	0.61
g	0.56	0.50	0.64	0.69	0.72	0.57	1.0	0.68
h	0.60	-	-	0.79	0.66	0.61	0.68	1.0

Figure 4.5 & Table 4.2: Graph based representation of sentences using semantic similarities >0.4 between sentences as edge weight and table of semantic similarity values >0.4

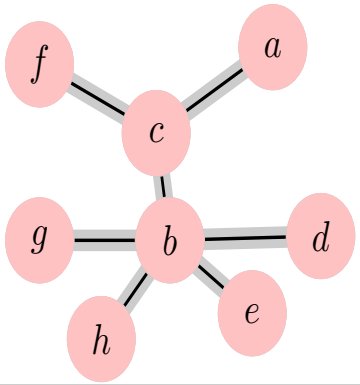
Figure 4.5. See Table 4.2 to observe every similarity value less than 0.4 has been discarded. After conducting experiments, section 5.4, we observe the system gives best predictions with graph-based features derived by similarities greater than the threshold of 0.4.

4.2 Graph-Based Features

We use semantic similarities represented as graph. The graph-based features give us more information about how an essay occupies the spatial space. To analyze the structural properties and patterns in a network, we explore graph characteristics as highlighted by Kolaczyk, Eric D [51, 52]. To obtain the semantic properties of an essay, the following are the graph-based features computed:

1) **Minimum spanning tree**: Minimum spanning tree is a subset of graph edges that connect all the vertices with minimum possible total weight [33]. When

an essay is represented as a graph in a semantic space, it depicts the semantic association between each sentence. The main motivation behind obtaining the minimum spanning tree is to derive the weakest similarity connections in the essay, Figure 4.6. This graph represents a minimum spanning tree for the fully connected graph in Figure 4.4. The weakest connections which span through the graph representation of an essay tells us about the minimum coherence, i.e., traversing through all the statements of the essay. We sum the values of the edges weights of the tree and get a minimum spanning tree sum. The listing in 4.1, we convert the semantic similarity results into a sparse matrix using python functionality from *scipy* [47]. This matrix is used to find the minimum spanning tree using the *minimum_ spanning_ tree* function from *scipy* again.



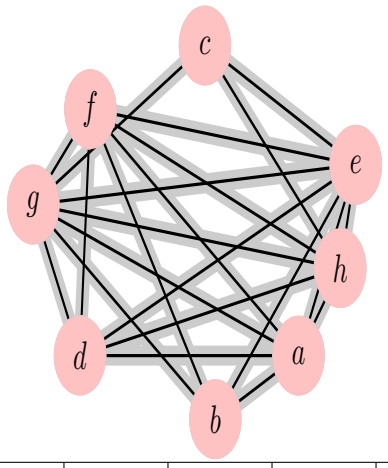
Sentences	a	b	c	d	e	f	g	h
a	-	-	0.19	-	-	-	-	-
b	-	-	0.34	0.35	0.46	-	0.50	0.25
c	0.19	0.34	-	-	-	0.28	-	-
d	-	0.35	-	-	-	-	-	-
e	-	0.46	-	-	-	-	-	-
f	-	-	0.28	-	-	-	-	-
g	-	0.50	-	-	-	-	-	-
h	-	0.25	-	-	-	-	-	-

Figure 4.6 & Table 4.3: Minimum spanning tree and table displaying the edges used in the tree

Listing 4.1: Obtaining minimum spanning tree from graph representing essay similarities

```
def minimum_spanning_tree_sum(semantic_results):
    csr_results = csr_matrix(semantic_results)
    #Using csr-matrix function from scipy.sparse
    min_st = minimum_spanning_tree(csr_results)
    #Using minimum spanning tree function from
      scipy.sparse
    mst_list = list (min_st)
    #Convert the minimum-spanning-tree edge
    values into a list for summation
    sum_min_st = sum(mst_list)
    return sum_min_st
```

2) **Maximum spanning tree**: Maximum spanning tree is a subset of graph edges that connect all the vertices with the maximum possible total weight of the edges. A minimum spanning tree with reciprocated edge weight (for instance, value of 0.56 in cell (a,b) in Table 4.1 is converted to 1.78 = (1/0.56) in Table 4.4) is a maximum spanning tree for the original edge weights. Thus, the inverted the values of Figure 4.4 are used to obtain this sub-graph, Figure 4.8. Each edge value is reciprocated, see Table 4.4. As the graph depicts the semantic association between each sentence, to find the most similar connection which binds all the sentences of a graph, we compute the maximum spanning tree. Summing up the non-reciprocated original values from this subset gives us the maximum spanning tree sum. The Listing 4.2 shows the code for maximum spanning tree sum.

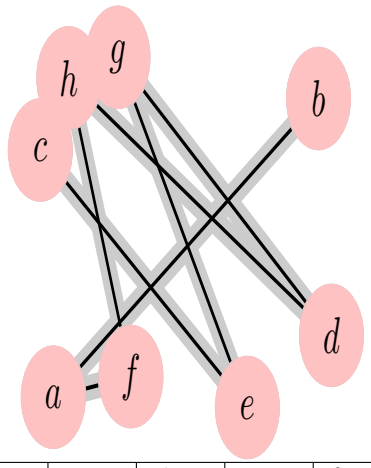


Sentences	a	b	c	d	e	f	g	h
a	1.0	1.78	5.26	1.694	1.785	1.315	1.785	1.66
b	1.785	1.0	2.941	2.857	2.173	1.818	2	4
c	5.263	2.941	1.0	1.408	1.923	3.571	1.562	3.125
d	1.694	2.857	1.408	1.0	2.040	1.666	1.449	1.265
e	1.785	2.173	1.923	2.040	1.0	1.754	1.388	1.515
f	1.315	1.818	3.571	1.666	1.754	1.0	1.754	1.639
g	1.785	2	1.562	1.449	1.388	1.754	1.0	1.470
h	1.666	4	3.125	1.265	1.515	1.639	1.470	1.0

Figure 4.7 & Table 4.4: Graph based representation of sentences using reciprocated semantic similarities between sentences as edge weight

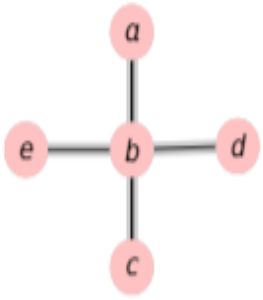
Listing 4.2: Obtaining maximum spanning tree from graph representing essay similarities

```
def maximum_spanning_tree_sum(semantic_results):
      csr_results = csr_matrix(semantic_results)
    #Using csr-matrix function from scipy.sparse
    max_st = maximum_spanning_tree(csr_results)
    #Using maximum-spanning-tree function
     from scipy.sparse
    max_list = list (max_st)
    #Convert into a list for summation
    sum_max_st = sum(max_list)
    return sum_max_st
```



Sentences	a	b	c	d	e	f	g	h
a	-	0.56	-	-	-	0.76	-	-
b	0.56	-	-	-	-	-	-	-
c	-	-	-	-	0.52	-	-	-
d	-	-	-	-	-	-	0.69	0.79
e	-	-	0.52	-	-	-	0.72	-
f	0.76	-	-	-	-	-	-	0.61
g	-	-	-	0.69	0.72	-	-	-
h	-	-	-	0.79	-	0.61	-	-

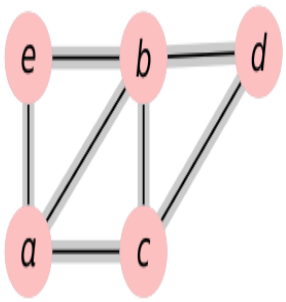
Figure 4.8 & Table 4.5: Maximum spanning tree and table displaying the original edge weight used in the tree



Node	Centrality
a	0.14
b	0.25
c	0.14
d	0.14
e	0.14
Average	0.162

Figure 4.9 & Table 4.6: Graph with 5 nodes and table showing centrality values for each node

3) **Closeness centrality**: Closeness centrality is measure of centrality in a network [36]. Closeness centrality for a node *a* in the graph is the inverse of the shortest path distances from node *a* to all *n-1* other nodes, while *n* being total number of



Node	Centrality
a	0.14
b	1.0
c	0.14
d	0.14
e	0.66
Maximum	0.416

Figure 4.10 & Table 4.7: Graph with 5 nodes and table showing centrality values for each node

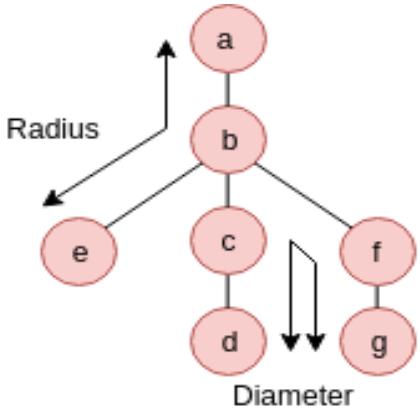
nodes in the graph. See Equation 4.1, *d(b,a)* represents shortest path distance between *a* and *b* and *n* is the number of nodes that can reach the node *a*. The distance *d* here represents the semantic similarity values as edge weights.

$$Closeness\_centrality(a) = \frac{1}{\sum_{b=1}^{n-1} d(b,a)} \tag{4.1}$$

For each graph based essay representation we find closeness centrality for each node. The Figure 4.9 shows a graph with 5 nodes with 4 edges and Table 4.6 displaying closeness centrality values when assumed that distance between each node is 1. The average centrality of all the nodes for this graph is 0.16. The Figure 4.10 and Table 4.7 show another graph with 5 nodes with 7 edges and the average closeness centrality of 0.41. We want to emphasize that the second graph has higher average centrality value, thus each node in the graph is more central and has more cohesion associated to the essay.

4) **Graph eccentricities**: The maximum distance between a vertex to all other vertices in the graph is called eccentricity of the vertex. The eccentricity of a node *a* is the maximum distance from *a* to all other nodes in the graph *G*. The graph in Figure 4.11, has seven nodes and the Table 4.8 displays eccentricity for each node. Eccentricities can be further studied as :

- Diameter: The diameter of a graph is the greatest distance between any pair of vertices. From all the eccentricities of the vertices in a graph, the diameter of the connected graph is the maximum of all those eccentricities. The diameter of a graph indicates how widespread the nodes are. A large diameter value indicates a widespread graph which is less cohesive [66]. In Figure 4.11, the maximum eccentricity values is 4 for node *d*. Thus, the diameter of the graph is 4.
- Radius: The minimum eccentricity from all the vertices is considered as the radius of the graph. From all the eccentricities of the vertices in a graph, the radius of the connected graph is the minimum of all those eccentricities. The minimum eccentricity for a node in Figure 4.11 is 2. Therefore, the radius of the graph is 2.



Node	Eccentricity
a	3
b	2
c	3
d	4
e	3
f	3
g	4

Figure 4.11 & Table 4.8: Graph with 5 nodes and table with node eccentricities

5) **Density difference**: Density of graph can be defined as the actual number of edges compared to the possible number of edges of that graph. A higher number of edges in the graph indicates a higher density. The Equation 4.2 is used to derive the density of a graph, m is the number of edges and n is the number of nodes. We compare the graph in Figure 4.4 with the graph in Figure 4.5 to compute the density difference before and after dropping all the similarities below 0.4. This difference highlights the number of weak connections with similarities less than 0.4 in the graph, higher density difference indicates

a high number of weak connections existed in the graph before we dropped them.

$$d = \frac{2m}{n(n-1)} \tag{4.2}$$

6) **Number of Central nodes**: Nodes with eccentricity equal to the radius (which is the minimum eccentricity possible between any pair of vertices) are referred to as central nodes. More central number of nodes implies a closely related compact graph. In Figure 4.11, only one node, i.e., *b* has eccentricity equal to the radius of the graph. Thus, the number of central nodes for this graph is 1.

7) **Significant words and their similarity**: As per existing researches in the literature, closeness between the essay and the question prompt is important [85]. We use TF-IDF to find important words in the essay and the prompt. *TF-IDF* stands for Term frequency - Inverse Data frequency. For each sentence in the essay, we compute the frequency of each term in the sentence that is called *TF*. *IDF* refers to the importance of the word's existence in a text, the words which occur rarely are weighed up and which occur too often are weighed down. [88]

$$tf(t, d) = f_{t,d} \tag{4.3}$$

In Equation 4.3, the term frequency *tf(t,d)* is the number of times that a term t appears in one sentence d.

$$idf(t, D) = \log \frac{|D|}{n_t} \tag{4.4}$$

In Equation 4.4, the inverse document frequency *idf(t, D)* looks at whether the term t is common or not across all the set of sentences in the essay D. n_t is the total number of sentences containing the term t.

$$\mathrm{tf-idf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D) \tag{4.5}$$

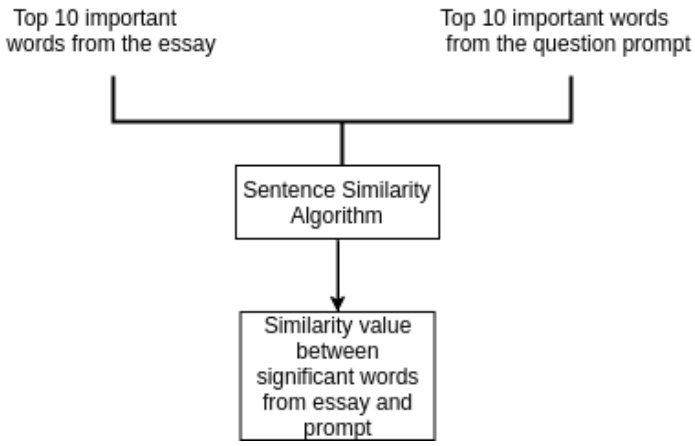


Figure 4.12: Finding significant words and computing their similarity

In equation 4.5, *tf-idf* shows the importance of the term t in a sentence d given the essay D. We filter nouns and verbs out of the essay and the question prompt and compute TF-IDF. We find important words for each sentence in the essay and consolidate them and then pick up top 10 from each essay based on their *tf-idf* scores. Top 10 words are picked based on the *TF-IDF* from both the essay and the question prompt. A limitation can occur if an essay prompt consists of less than 10 words which can be handled by making changes in the code. These two-word sets are compared to each other using the semantic similarity algorithm with fastText word embeddings which gives a similarity value for both the lists. Figure 4.12 is a chart showing the process of similarity derivation between top-words from essay and prompt.

8) **Similarity between the prompt and the entire essay**: Semantic similarity between the entire essay and the question prompt is computed to find how meaningfully similar they are. Since every essay is written as a response to a question prompt, we believe they should display semantic similarity between them. We use the semantic similarity algorithm as described in Section 4.1.1 to compute semantic similarity between the essay response and question prompt.

Chapter 5

Implementation and Evaluation

5.1 Data

To validate and compare our results to the exsiting systems, we use data from the Automated Student Assessment Prize (ASAP) [1] competition sponsored by the William and Flora Hewlett Foundation (Hewlett) held in 2012. The dataset is composed of 8 different data-sets of different genres which are argumentative (Arg), responsive (Res), narrative (Nar), persuasive (Per) and expository (Exp). Each data-set is a collection of responses to its own prompt. Students from grade 7 to grade 10 have written the essays ranging from 150 to 550 words per essay, refer Table 5.1. A minimum of 2 human graders has provided the grades which are available to us.

Set	Essays	Genre	Avg. Length	Score
1	1,783	Arg	350	2-12
2	1800	Arg	350	1-6
3	1,726	Res	150	0-3
4	1,772	Res	150	0-3
5	1,805	Res	150	0-4
6	1,800	Res	150	0-4
7	1,569	Nar/Per/Exp	250	0-30
8	723	Nar	650	0-60

Table 5.1: ASAP data-set

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

5.2 Evaluation Metric

The evaluation metric used to test our system is Quadratic Weighted Kappa (QWK), as this was the official evaluation metric chosen by the ASAP competition. QWK is a measure of agreement between two raters. In case of essay evaluation system it is the agreement between predicted score by the system and the grade by human rater. QWK ranges between values 0 (no agreement) to 1 (complete agreement) [6], refer Table 5.2. Each dataset E has N number of possible ratings. Every essay in each data-set can be indicated by a tuple (ea, eb) where ea refers to the human rater's score and eb refers to the predicted score by the evaluation system. An N-by-N histogram matrix O is constructed over the essay ratings, where Oi,j refers to the number of essays with grade i by human grader and a grade j by prediction system. An N-by-N matrix of weights w, is calculated based on the difference between rater's scores refer Equation 5.1 and Quadratic Weighted Kappa k is found by the Equation 5.2.

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2} \tag{5.1}$$

$$k = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \tag{5.2}$$

QWK	Agreement strength
<0.2	Poor
0.21-0.40	Fair
0.41-0.60	Moderate
0.61-0.80	Good
0.81-1.00	Very Good

Table 5.2: QWK value interpretations

5.3 Feature Selection

Feature selection is the core building block of a machine learning model and has a huge impact on the performance. Moderately performing or irrelevant features can negatively affect the prediction. To make the best use of a machine learning model, the foremost significance should be given to feature selection and data pre-processing.

Even a single bad feature can hamper the model's performance. Some key benefits of feature selection are:
1) Reduces over-fitting: Getting rid of some features helps to get rid of redundant data. Thus the machine learning model's prediction is not based on any noisy data.
2) Reduces training time: Fewer features means less complexity and lesser training time for the algorithm.
3) Enhances the performance of the prediction model: When only the best predicting features are fed to the model, the modeling accuracy improves.

5.3.1 Methodologies

To select the best features we perform selection techniques [41] mentioned below:

1) **Univariate selection**: We use a univariate linear regression test by sci-kit-learn [84]. A linear model is used to derive the effect of each feature (regressor) on the predictability of the model. Univariate feature selection works by selecting the best predictive features based on statistical tests. As we are solving regression problem here we use the function *f_ regression* with *sklearn.feature_ selection* from skicit learn library in python [84].

2) **Recursive feature elimination**: Important features are selected by recursively removing features and testing the prediction with remaining ones to find which features have with most predictive power. This is done using *featureimportances* function by *skicit-learn* [84] in python.
In Table 5.3, the second and third column displays the results from Univariate linear regression test and ranks from Recursive feature elimination respectively.

3) **QWK vs. Features**: As we are trying to minimize the number of features for better prediction, it is important to decide the minimum number of features possible without compromising the performance of the model. As per the ranking obtained by recursive feature elimination, we keep adding features to the model starting from the top ranking feature, i.e., the number of characters and recursively adding second, third and so on till the last one, i.e., the diameter of the graph. We derive the QWK value after each iteration of adding a new feature to the model for each of the eight

data-sets. A graph is plotted using the average QWK values over all data-sets vs. the number of features used towards obtaining it, Figure 5.1. We see the top 23 ranked features from the recursive feature elimination technique gives us the best performance with highest QWK value. Therefore, we use the top 23 features from the Table 5.3 for our final prediction model.

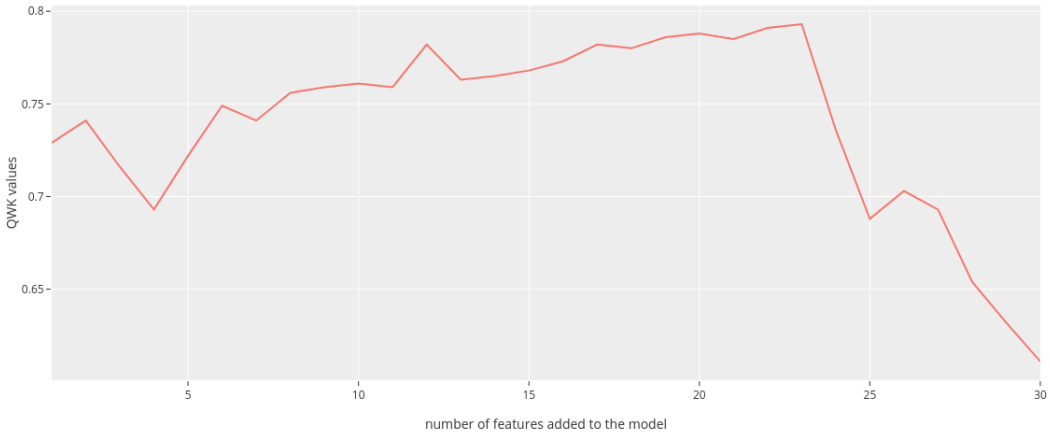


Figure 5.1: QWK values vs number of features recursively added to the model

4) **Predictability of each feature**: ASAP data-set has eight data-sets of different genres. Therefore it is important to test how the individual features behave in terms of prediction over different data-sets. We run our prediction model for each feature individually over each data-set recursively and obtain a QWK value for each. This helps us to validate the above-mentioned feature elimination techniques and help us select the features which perform well over each data-set. Fig. 5.2 shows the top 23 features selected for the model have a good performance in term of prediction power on each of the data-sets.

Feature name	**Univariate selection**	**Ranking from recursive feature elimination**
Number of characters	3.12E+02	1
Number of words	2.91E+02	2
Semantic similarity between top-words of essay and prompt	2.20E+02	3
Sum of maximum spanning tree	1.67E+02	4
Sum of minimum spanning tree	1.69E+02	5
Density difference between original and reduced graph of similarities	1.63E+02	6
Common sentence length	1.63E+00	7
Unique part of speeches used	1.58E+02	8
Words ending with -ing	1.25E+02	9
Misspelled words	1.25E+02	10
Number of singular subjects	2.07E+02	11
Closeness centrality	1.20E+02	12
Number of plural subjects	8.84E+01	13
Number of co-ordinating conjunctions	8.29E+01	14
Past tense verb	7.21E+01	15
Verb, base-form	6.62E+01	16
Verb, present participle	8.61E+01	17
Verb, non-3rd person singular present	8.84E+01	18
Positive polarity	7.59E+01	19
Verb, past participle	8.29E+01	20
Negative polarity	4.12E+01	21
Number of most frequent repetitive word	2.22E+01	22
Neutral polarity	6.19E+00	23
Semantic similarity to question prompt	2.09E+01	24
Number of superlative adjectives	2.09E+00	25
Number of predeterminants	1.65E-01	26
Number of exesntial-there	3.52E-01	27
Radius	2.68E-01	28
Central nodes	8.80E+00	29
Diameter	8.80E+00	30

Table 5.3: Results from feature selection techniques

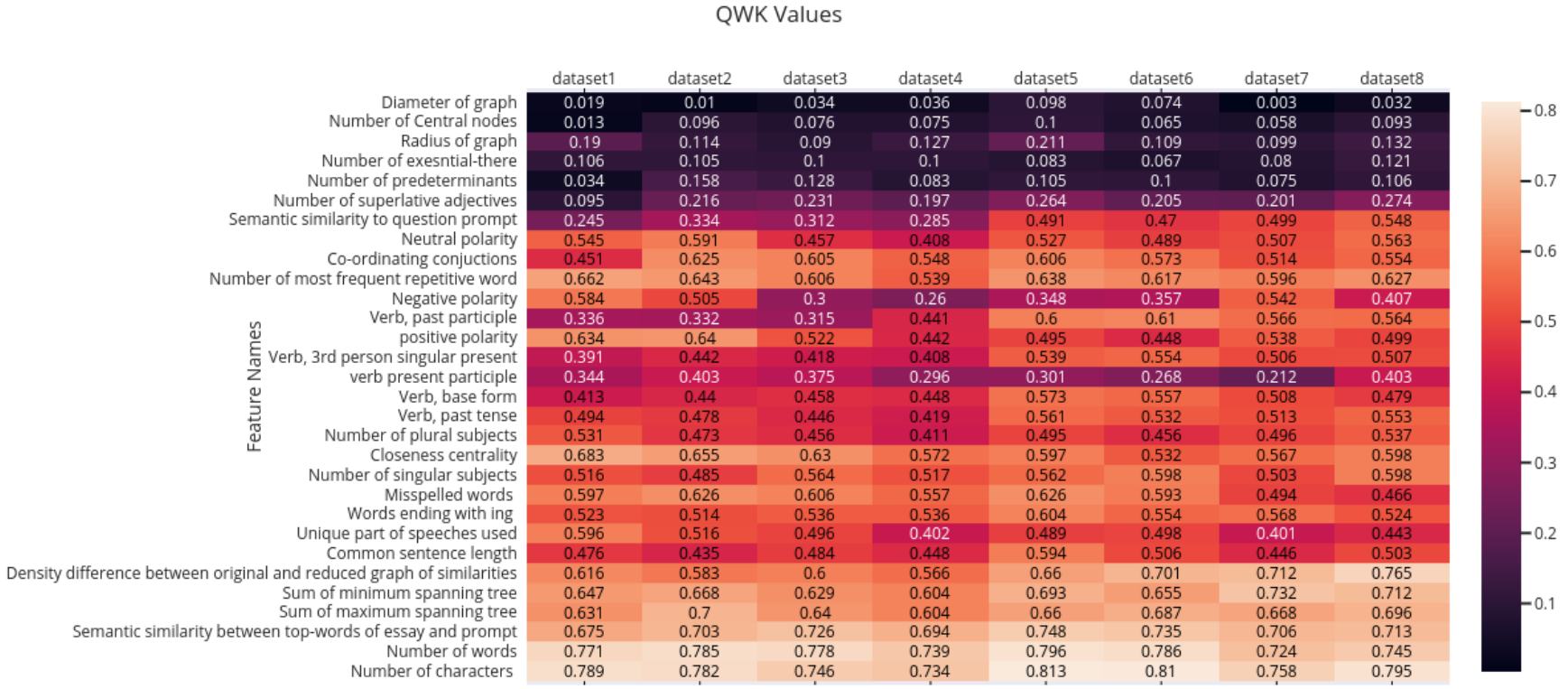


Figure 5.2: Heatmap of QWK values for each feature over the 8 data-sets

5.3.2 Conclusion

Based on the results mentioned above, we selected the top 23 best performing features for our prediction model. We obtain results from univariate feature selection and ranking from the recursive feature elimination technique. To validate these techniques, we use the QWK values for additional assessment. We plot a graph of QWK values vs. the number of features which are recursively added to the model based on ranks from recursive feature elimination. This plot shows the model performs the best when the top 23 features are used. Additionally, to analyze each feature's prediction power we recursively run the model with just one feature at a time over all data-sets and obtain a heatmap to identify unsatisfactorily performing features. This heatmap validates the performance of our selected top 23 features. Thus, they are added to the prediction model.

5.4 Threshold for Graph-Based Features

As we discussed earlier in Chapter 4 about graph-based feature extraction from semantic similarities in the essay after dropping the similarities less than a certain threshold. In this section, we elaborate on the selection of this threshold value.
For graph-based features like diameter, radius and closeness centrality it is important to use only the strong semantic relations as even one weak connection can affect the predictions. We checked how the system's predictability behaves with graph-based features including all edges compared to the predictability after dropping edges with different thresholds values.
We obtain different sets of graph based features with semantic similarities greater than threshold values starting from 0.1 till 0.9. We test our prediction model with these features and find QWK value for every data-set for each set of features. We found that feature set obtained with threshold of 0.4 gives us the best prediction values, Table 5.4.

5.5 Prediction Models

We treat score prediction as a regression problem and use supervised learning for it. We chose a various learning algorithms and train the scored essays by humans and then provide it with features associated with unseen essays to obtain the predicted

Threshold	D-1	D-2	D-3	D-4	D-5	D-6	D-7	D-8	Avg.
0.1	0.79	0.79	0.56	0.59	0.85	0.69	0.68	0.70	0.71
0.2	0.78	0.63	0.78	0.55	0.77	0.66	0.70	0.65	0.69
0.3	0.80	0.77	0.62	0.70	0.78	0.80	0.71	0.61	0.72
0.4	**0.83**	**0.81**	**0.77**	**0.72**	**0.87**	**0.82**	**0.75**	**0.78**	**0.79**
0.5	0.79	0.69	0.66	0.66	0.77	0.78	0.74	0.73	0.73
0.6	0.88	0.60	0.61	0.63	0.78	0.66	0.74	0.63	0.69
0.7	0.87	0.60	0.60	0.58	0.76	0.84	0.74	0.61	0.70
0.8	0.78	0.60	0.61	0.55	0.75	0.68	0.73	0.63	0.67

Table 5.4: Comparison of results considering different threshold to derive semantic similarities

score using the regression function. The technologies and libraries used for prediction models are sci-kit-learn [84], Keras [34], and Numpy [75]. As there are many supervised regression models available, it is crucial to find which one suits best for our problem and yields good results. The three prediction models we tried:

1) Support Vector Machine: The method provided by support vector machines to handle regression problems is called support vector regression [100]. A kernel function is a method to solve the non-linear problem by a linear classifier. We use the sigmoid kernel, which is equal to a using a two-layer, perceptron neural network [103] and found to perform well in regression problems [9]. We use the SVR function from the sk-learn library in python to run this prediction model. Refer Table 5.5 for parameters used.

Parameter	Value
kernel	sigmoid

Table 5.5: Support vector regressor parameters

2) Random Forest Regressor: Random forest is ensemble of multiple decision trees. Rather than depending on a single decision tree, it depends on various decision trees on sub-samples of the data-set [57]. We use the *RandomForestRegressor* method from the sk-learn library in python to run this prediction model. Refer Table 5.6 for the value of parameters used in the model. After running experiments with several combinations of parameter values, it was found the model worked the best with the mentioned values in Table 5.6.

Parameter	Value
n_estimator	1000
random_state	42

Table 5.6: Random forest regressor parameters

3) Three layer neural network: We use Keras API which runs high-level deep neural networks [34]. The neural network consist of three layers with 23 inputs, 2 hidden layers and one output layer:

- Input layer: This layer which feeds the input data to the model. The input layer has the same number of neurons as input attributes, i.e., 23 in our case.
- Hidden layer: These layers use *backpropagation* to optimize the weights of the input variables thus improving the prediction power. We use two hidden layers in our model. As per experiments conducted we observed that adding anymore number of layers to the model do not help improve the results any further; therefore, we use two hidden layers.
- Output layer: This layer gives the final output based on inputs and the hidden layers.

We standardize our data as they all vary in their scale. Refer table 5.7 for other parameters used in running the model.

Parameter	Value
input_dim	23
kernel_initializer	normal
activation	relu
optimizer	adam
loss	mean_squared_error
epochs	100
batch_size	50

Table 5.7: Three layer neural network parameters

We run these three prediction models over all the eight data-sets from ASAP. Table 5.9 shows the QWK scores obtained over different data-sets with these models. We observe using three layered neural network in scikit-learn with the KerasRegressor

Feature Set	**Average QWK over 8 data sets**
Syntax	0.701
Semantic	0.653
Sentiment	0.392
Syntax with Semantic	0.741
Syntax with Sentiment	0.744
Semantic with Sentiment	0.679
Syntactic, Sentiment and Semantic	0.793

Table 5.8: Effect of syntactic, semantic and sentiment based features

Learning Models	**D-1**	**D-2**	**D-3**	**D-4**	**D-5**	**D-6**	**D-7**	**D-8**
SVM (sigmoid kernel)	0.78	0.55	0.61	0.64	0.69	0.61	0.73	0.65
Random forest	0.77	0.54	0.57	0.51	0.75	0.61	0.71	0.66
Three layer neural network	0.83	0.81	0.77	0.72	0.87	0.82	0.75	0.78

Table 5.9: Comparison of QWK for different supervised learning models in score prediction

class wrapper performs the best in terms of score predictions. Thus, we choose the three layered neural network as our final prediction models for our system's results.

5.6 Effect of Syntactic, Semantic and Sentiment based features

As the title of the thesis suggest, we propose to use the syntactic, semantic and sentiment based features together to help automate the process of essay's score predictions. To justify their use together, we conducted experiments using these three set of features separately, using two sets at a time and then all three together. We observe when we use all three sets of these features together our system performs the best in terms of score predictions. We run the selected prediction model over all 8 data-sets and average the performance of each set of features by obtaining QWK values, Table 5.8 shows how the average QWK changes over different combinations of features. We get the best results when all three set of features i.e., syntactic, semantic and sentiment are used together.

5.7 Discussions and Results

To evaluate the automated evaluation system's predictive power, QWK for the predicted scores of each data-set is calculated. Only the top 23 most significant features are used, Table 5.3. Treating score prediction as a regression problem, we use a Keras based regression model on an i5 processor with 16GB RAM and 1050 Ti graphics card. We divide the data into using a ratio of 75:25 and set aside the 25% for validation of the model later. We use 75% of the data for training the model with ten-fold cross-validation to validate the model's performance [95]. We use the remaining 25% which is never seen by the model during training to test the score predictions by the trained model. We tested the model for each of the eight data-sets to obtain a QWK value. We conducted experiments with different parameter values for neural networks and found the best results when the number of epochs is equal to 100, and the batch size is equal to 50 for the regression model. Table 5.10 shows the comparison between QWK values over each data-set from our proposed system compared to popular existing automated scoring systems. The results of other systems being compared were obtained from literature [18,45,85,97] and Kaggle's website. The pro-

posed system gives improved results with a fewer number of features involved, while covering all the necessary aspects of language processing, making it very reliable for essay grading with uniform assessment thoroughly. Our system uses only 23 features, which is significantly less than the number used by all the other systems in comparison which reduces the noise during model training, refer Table 5.10. Our system performs better than any other system in four out of eight data-sets in comparison, with an average QWK of 0.793. The only system outperforming our system in the remaining four data-sets is SBLSTMA [18] with average QWK value of 0.801, i.e., only 0.8% better than our system. SKIPFLOW [110] and SBLSTMA uses 14 main features, plus many more sub-features, which have not been mentioned explicitly in the published research, thus, making the research non-reproducible and making it hard to make a comparison. We try to contact the authors to provide information about sub-features used in their research, but there was no response. The system Tpioc-BiLSTM-attention [19] which works via hierarchical recurrent model does not provide any details about the features used in their published research. We also want to emphasize that extraction of a massive number of features vs. 23 features adds to time complexity as well. We also compared our system to work published recently in 2018; the system is called TDNN [45] which uses a two-layer neural network to reach an average QWK score of 0.7365, i.e., 7.1% lesser than us. Our system uses the minimum number of features compared to existing systems with better results.

System	No. of Features	D-1	D-2	D-3	D-4	D-5	D-6	D-7	D-8	Avg.
The Proposed System	**23**	0.83	**0.81**	0.77	0.72	**0.87**	**0.82**	0.75	**0.78**	0.793
SBLSTMA	14 plus several sub-features	**0.86**	0.73	**0.78**	**0.82**	0.84	0.82	0.81	0.75	0.80
SVMrank	33	0.80	0.68	0.67	0.73	0.80	0.71	0.77	0.71	0.73
SKIPFLOW	14 plus several sub-features	0.83	0.68	0.69	0.79	0.81	0.810	0.800	0.69	0.76
Tpioc-BiLSTM-attention	Not mentioned	0.82	0.69	0.69	0.81	0.81	0.82	0.80	0.70	0.77
e-rater	46	0.82	0.69	0.72	0.80	0.81	0.75	**0.81**	0.70	0.77
IntelliMetric	400	0.78	0.68	0.73	0.79	0.83	0.76	**0.81**	0.68	0.76
BLRR	15 plus several sub-features	0.76	0.60	0.62	0.74	0.78	0.77	0.73	0.62	0.70
TDNN	Not clear	0.76	0.68	0.62	0.75	0.73	0.67	0.65	0.57	0.68

Table 5.10: Comparison of results with existing systems

Chapter 6

Conclusion

Our proposed system successfully incorporates not only the rule-based grammar and syntax checks, but also the semantic similarities within the essay depicting its coherence. We propose to use semantics based graph based relationships within the essay content and polarity of opinion expressions. We incorporate pragmatic syntax features, semantic features depicting coherence based on accurate semantic similarity values, and sentiment-based polarity features. Thus, our research will help to reduce the number of independent features needed to be extracted from the text while utilizing the most vital features needed in automated essay evaluation for better prediction accuracy. Lesser is the number of features lesser is the redundant data; thus, predictions are not over-fitted. Our work not only provides accuracy values but also provides details to the readers making it reproducible. As compared to other existing systems, our work is more transparent and repeatable. Thus, this research can help eradicate the hours of manual work for teachers, giving them the freedom to concentrate more on academic teaching and learning and also helps give students the assurance of fair and consistent assessment throughout every submission.

6.1 Future Work

There are other machine learning models which can be explored and tested with the proposed system in this research. Models like LSTM [53] seem promising for sequential data if work is done to derive the correlation between essay scores in the data-set LSTM can give good results. Existing researchers state that semantics of a coherent essay changes gradually through its text [28]. This approach encourages

comparison of consecutive sentences in the essay to study the information flow. Many times, as a matter of writing style, abrupt shifts between consecutive sentences are used to convey information in successive sentences [22]. To study the coherence and flow of information in an essay, a comparison between sentences can be done using a sliding window frame. An optimized value for the sliding window can be obtained, and the sentences which happen to fall within the window can be compared for semantic similarity to assess the value of information. We believe this model lacks the study of ontology-based connections in the essay's text and careful extraction of ontology-based features can be beneficial [76]. The model can be further improved by including other types of centrality based features existing in the graph networks [55].

Appendix A

Additional Information

Following is an essay from the ASAP dataset with sentences given below from A to H , followed by the sentence similarities found between them using sentence similarity algorithm, refer table A.1

A - I don't think that computers is the most positive effect.

B - Because most computer's are not like that the computers are not the smartest but it does help you when you need to do something like a project when you have to look for pictures and when you need help with math you can use the calculator on the computer but the computer does not help you on all your subject's the computer only help's you on a couple of your subject's that is the main reason I use the computer.

C - Sometimes when I'm doing homework I let the computer do it for me so it is eaiser for me to do and I look for website's on it they give you really good website's that is why mostly everyone use's the computer they mostly use the comptuer to look for job's, car's, or a house that is what they mainly use the computer for most people do not like useing the internet because they think it's cheating to use the computer.

PAIR	VALUE
A-B	0.43639
A-C	0.434993
B-C	0.798103

Table A.1: Results of semantic similarities between sentences of an essay

Figure A.1 shows the above mentioned essay in graph based form. Since, the essay has three sentences with semantic similarities between them greater than 0.4; therefore, none of the edges are dropped. Figure A.2 represents maximum spanning

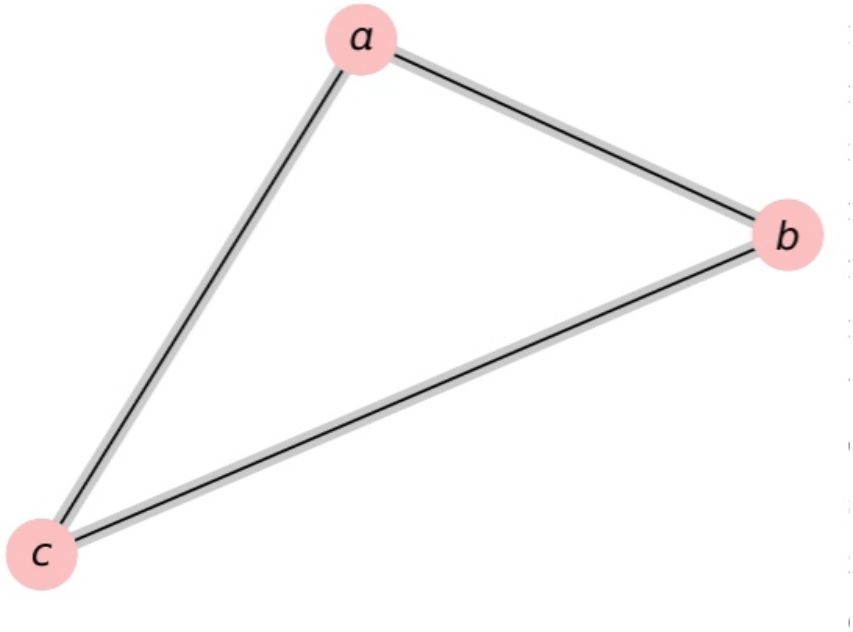


Figure A.1: A three node graph depicting semantically connected essay sentences

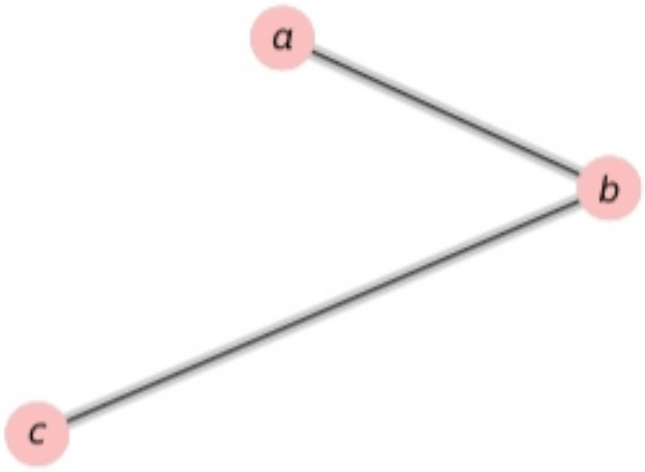


Figure A.2: Maximum spanning tree

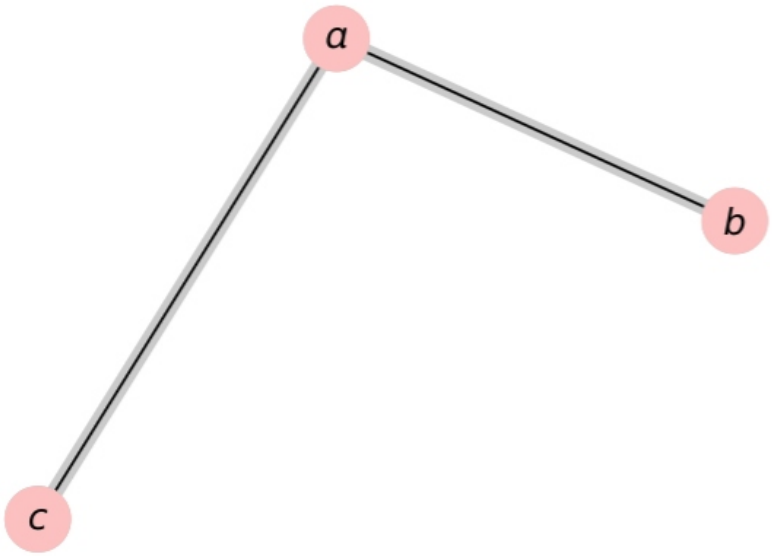


Figure A.3: Minimum spanning tree

tree for the essay which includes the edges A-B and B-C with maximum weight i.e semantic similarity. Figure A.3 represents minimum spanning tree for the essay which includes the edges A-B and A-C with minimum weight i.e semantic similarity.

Appendix B

List of Abbreviations

- **AES** Automated essay scoring
- **ASAP** Automated student assessment prize
- **QWK** Quadratic weighted kappa
- **NLTK** Natural language toolkit
- **IEA** Intelligent essay assessor
- **NLP** Natural language processing
- **LSA** Latent semantic analysis
- **LDA** Latent dirichlet allocation
- **AI** Artificial intelligence
- **TF** Term frequency
- **IDF** Inverse data frequency
- **VADER** Valence aware dictionary and sentiment Reasoner
- **SVM** Support vector machine
- **API** Application program interface

Bibliography

[1] Khurshid Ahmad. *Affective computing and sentiment analysis: Emotion, metaphor and terminology*, volume 45. Springer Science & Business Media, 2011.

[2] Keith Allan. A note on the source of there in existential sentences. *Foundations of language*, 7(1):1–18, 1971.

[3] Gregory V Bard. Spelling-error tolerant, order-independent pass-phrases via the damerau-levenshtein string-edit distance metric. In *Proceedings of the fifth Australasian symposium on ACSW frontiers-Volume 68*, pages 117–124. Citeseer, 2007.

[4] Beata Beigman Klebanov, Jill Burstein, and Nitin Madnani. Sentiment profiles of multiword expressions in test-taker essays: The case of noun-noun compounds. *ACM Transactions on Speech and Language Processing (TSLP)*, 10, 07 2013.

[5] Beata Beigman Klebanov, Jill Burstein, Nitin Madnani, Adam Faulkner, and Joel Tetreault. Building subjectivity lexicon(s) from scratch for essay data. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, pages 591–602, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[6] Kenneth J Berry, Janis E Johnston, and Paul W Mielke Jr. Weighted kappa for multiple raters. *Perceptual and motor skills*, 107(3):837–848, 2008.

[7] Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. Measuring semantic similarity between words using web search engines. *www*, 7:757–766, 2007.

[8] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8, 2011.

[9] Sabri Boughorbel, J-P Tarel, and Nozha Boujemaa. Conditionally positive definite kernels for svm based image recognition. In *2005 IEEE International Conference on Multimedia and Expo*, pages 113–116. IEEE, 2005.

[10] Sahib Singh Budhiraja and Vijay Mago. Extracting learning outcomes using machine learning and white space analysis. In *Proceedings of the 4th EAI International Conference on Smart Objects and Technologies for Social Good*, pages 7–12. ACM, 2018.

[11] Michael Buhrmester, Tracy Kwang, and Samuel D. Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5, 2011. PMID: 26162106.

[12] Jill Burstein. The e-rater® scoring engine: Automated essay scoring with natural language processing. 2003.

[13] Jill Burstein, Beata Beigman-Klebanov, Nitin Madnani, and Adam Faulkner. 17 automated sentiment analysis for essay evaluation. *Handbook of Automated Essay Evaluation: Current Applications and New Directions*, page 281, 2013.

[14] Jill Burstein, Claudia Leacock, and Richard Swartz. Automated evaluation of essays and short answers. 2001.

[15] Nigel P Chapman. *LR parsing: theory and practice*. CUP Archive, 1987.

[16] Clinton I Chase. Essay test scores and reading difficulty. *Journal of Educational Measurement*, pages 293–297, 1983.

[17] Michel Chein and Marie-Laure Mugnier. *Graph-based knowledge representation: computational foundations of conceptual graphs*. Springer Science & Business Media, 2008.

[18] Hongbo Chen, Ben He, Tiejian Luo, and Baobin Li. A ranked-based learning approach to automated essay scoring. In *2012 Second International Conference on Cloud and Green Computing*, pages 448–455. Ieee, 2012.

[19] M. Chen and X. Li. Relevance-based automated essay scoring via hierarchical recurrent model. In *2018 International Conference on Asian Language Processing (IALP)*, pages 378–383, Nov 2018.

[20] Yanqing Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. The expressive power of word embeddings. *arXiv preprint arXiv:1301.3226*, 2013.

[21] Steve Y Chiang. Assessing grammatical and textual features in l2 writing samples: The case of french as a foreign language. *The Modern Language Journal*, 83(2):219–232, 1999.

[22] Pierre Coirier and Caroline Golder. Writing argumentative text: A developmental study of the acquisition of supporting structures. *European Journal of Psychology of Education*, 8(2):169–181, Jun 1993.

[23] Saadiyah Darus and Kaladevi Subramaniam. Error analysis of the written english essays of secondary school students in malaysia: A case study. *European journal of social sciences*, 8(3):483–495, 2009.

[24] Semire Dikli. An overview of automated scoring of essays. *The Journal of Technology, Learning and Assessment*, 5(1), 2006.

[25] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479, 2004.

[26] Noura Farra, Swapna Somasundaran, and Jill Burstein. Scoring persuasive essays using opinions and their targets. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 64–74, 2015.

[27] Peter W. Foltz, Walter Kintsch, and Thomas K Landauer. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25(2-3):285–307, 1998.

[28] Peter W. Foltz, Walter Kintsch, and Thomas K Landauer. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25(2-3):167–184, 1998.

[29] Michael Thomas Gentner. The impact of coordinating conjunction use on the sentence development of thai and khmer university student writers. *Panyapiwat Journal*, 8(3):178–187, 2016.

[30] Philippe Giabbanelli and Andrew Tawfik. Overcoming the pbl assessment challenge: Design and development of the incremental thesaurus for assessing causal maps (itacm). *Technology, Knowledge and Learning*, 09 2017.

[31] Philippe J. Giabbanelli, Andrew A. Tawfik, and Vishrant K. Gupta. *Learning Analytics to Support Teachers' Assessment of Problem Solving: A Novel Application for Machine Learning and Graph Algorithms*, pages 175–199. Springer International Publishing, Cham, 2019.

[32] Cliff Goddard. *Semantic analysis: A practical introduction*. Oxford University Press, 2011.

[33] Ronald L Graham and Pavol Hell. On the history of the minimum spanning tree problem. *Annals of the History of Computing*, 7(1):43–57, 1985.

[34] Antonio Gulli and Sujit Pal. *Deep Learning with Keras*. Packt Publishing Ltd, 2017.

[35] Vishrant K. Gupta, Philippe J. Giabbanelli, and Andrew A. Tawfik. An online environment to compare students' and expert solutions to ill-structured problems. In Panayiotis Zaphiris and Andri Ioannou, editors, *Learning and Collaboration Technologies. Learning and Teaching*, pages 286–307, Cham, 2018. Springer International Publishing.

[36] Per Hage and Frank Harary. Eccentricity and centrality in networks. *Social networks*, 17(1):57–63, 1995.

[37] Zhen He, Shaobing Gao, Liang Xiao, Daxue Liu, Hangen He, and David Barber. Wider and deeper, cheaper and faster: Tensorized lstms for sequence learning. In *Advances in neural information processing systems*, pages 1–11, 2017.

[38] Andrew Heppner, Atish Pawar, Daniel Kivi, and Vijay Mago. Automating articulation: Applying natural language processing to post-secondary credit transfer. *IEEE Access*, 7:48295–48306, 2019.

[39] Derrick Higgins, Jill Burstein, Daniel Marcu, and Claudia Gentile. Evaluating multiple aspects of coherence in student essays. In *HLT-NAACL 2004*, 2004.

[40] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

[41] Jianping Hua, Waibhav D Tembe, and Edward R Dougherty. Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recognition*, 42(3):409–424, 2009.

[42] Thomas N Huckin and Linda Hutz Pesante. Existential there. *Written Communication*, 5(3):368–391, 1988.

[43] Clayton J Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*, 2014.

[44] Nitin Madnani Adam Madnani Jill Burstein, Beata Beigman-Klebanov. *Automated Sentiment Analysis for Essay Evaluation*. Routledge, 2013.

[45] Cancan Jin, Ben He, Kai Hui, and Le Sun. Tdnn: a two-stage deep neural network for prompt-independent automated essay scoring. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1088–1097, 2018.

[46] Wei Jin and Rohini K. Srihari. Graph-based text representation and knowledge discovery. pages 807–811, 01 2007.

[47] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001.

[48] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

[49] Tuomo Kakkonen, Niko Myller, Erkki Sutinen, and Jari Timonen. Comparison of dimension reduction methods for automated essay grading. *Journal of Educational Technology & Society*, 11(3):275–288, 2008.

[50] Beata Beigman Klebanov, Nitin Madnani, Jill Burstein, and Swapna Somasundaran. Content importance models for scoring writing from sources. In

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), volume 2, pages 247–252, 2014.

[51] Eric D Kolaczyk. Descriptive analysis of network graph characteristics. In *Statistical Analysis of Network Data*, pages 1–44. Springer, 2009.

[52] Eric D Kolaczyk and Gábor Csárdi. Descriptive analysis of network graph characteristics. In *Statistical analysis of network data with R*, pages 43–67. Springer, 2014.

[53] Jitendra Kumar, Rimsha Goomer, and Ashutosh Kumar Singh. Long short term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters. *Procedia Computer Science*, 125:676 – 682, 2018. The 6th International Conference on Smart Computing and Communications.

[54] Thomas K Landauer, Danielle S McNamara, Simon Dennis, et al. *Handbook of latent semantic analysis*. Psychology Press, 2013.

[55] Eric A. Lavin, Philippe J. Giabbanelli, Andrew T. Stefanik, Steven A. Gray, and Robert Arlinghaus. Should we simulate mental models to assess whether they agree? In *Proceedings of the Annual Simulation Symposium*, ANSS '18, pages 6:1–6:12, San Diego, CA, USA, 2018. Society for Computer Simulation International.

[56] Yuhua Li, David McLean, Zuhair A Bandar, Keeley Crockett, et al. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge & Data Engineering*, (8):1138–1150, 2006.

[57] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[58] Marina Litvak and Mark Last. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24. Association for Computational Linguistics, 2008.

[59] Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

[60] Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[61] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[62] Susan M Lottridge, E Matthew Schulz, and Howard C Mitzel. Using automated scoring to monitor reader performance and detect reader drift in essay scoring. *Handbook of Automated Essay Evaluation: Current Applications and New Directions, MD Shermis and J. Burstein, Eds. New York: Routledge*, pages 233–250, 2013.

[63] Danielle S McNamara, Scott A Crossley, and Philip M McCarthy. Linguistic features of writing quality. *Written communication*, 27(1):57–86, 2010.

[64] J. Burstein (Eds.) M.D. Shermis. Implementation and applications of the intelligent essay assessor. pages 68–88, 2013.

[65] Charles T Meadow, Bert R Boyce, and Donald H Kraft. *Text information retrieval systems*, volume 20. Academic Press San Diego, CA, 1992.

[66] Olena Medelyan. Computing lexical chains with graph clustering. In *Proceedings of the ACL 2007 student research workshop*, pages 85–90, 2007.

[67] Donald Michie, David J Spiegelhalter, CC Taylor, et al. Machine learning. *Neural and Statistical Classification*, 13, 1994.

[68] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[69] Tristan Miller. Essay assessment with latent semantic analysis. *Journal of Educational Computing Research*, 29(4):495–512, 2003.

[70] Eleni Miltsakaki and Karen Kukich. Automated evaluation of coherence in student essays. In *Proceedings of LREC 2000*, 2000.

[71] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[72] Ayman Mohamed Mostafa. An evaluation of sentiment analysis and classification algorithms for arabic textual data. *Int. J. Comput. Appl*, 158(3), 2017.

[73] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[74] Linda S. Norton. Essay-writing: what really counts? *Higher Education*, 20(4):411–442, Dec 1990.

[75] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[76] Nathan Ong, Diane J. Litman, and Alexandra Brusilovsky. Ontology-based argument mining and automatic essay scoring. 06 2014.

[77] Ellis B Page. The imminence of... grading essays by computer. *The Phi Delta Kappan*, 47(5):238–243, 1966.

[78] Ellis B Page. The use of the computer in analyzing student essays. *International review of education*, 14(2):210–225, 1968.

[79] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

[80] Ajay Patel, Alexander Sands, Chris Callison-Burch, et al. Magnitude: A fast, efficient universal vector embedding utility package. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 120–126, 2018.

[81] Atish Pawar, Sahib Budhiraja, Daniel Kivi, et al. Are we on the same learning curve: Visualization of semantic similarity of course objectives. *arXiv preprint arXiv:1804.06339*, 2018.

[82] Atish Pawar and Vijay Mago. Similarity between learning outcomes from course objectives using semantic analysis, blooms taxonomy and corpus statistics. *CoRR*, abs/1804.06333, 2018.

[83] Atish Pawar and Vijay Mago. Challenging the boundaries of unsupervised learning for semantic similarity. *IEEE Access*, 7:16291–16308, 2019.

[84] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[85] Peter Phandi, Kian Ming A Chai, and Hwee Tou Ng. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 431–439, 2015.

[86] Kenneth Lee Pike, Evelyn G Pike, and Kenneth Lee Pike. *Grammatical analysis*. Summer Institute of linguistics Dallas, Texas, 1977.

[87] Venkata Sai Sriram Pillutla. Helping users learn about social processes while learning from users: developing a positive feedback in social computing. 2017.

[88] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ, 2003.

[89] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[90] Vassiliki Rentoumi, George Giannakopoulos, Vangelis Karkaletsis, and George A Vouros. Sentiment analysis of figurative language using a word sense disambiguation approach. In *Proceedings of the International Conference RANLP-2009*, pages 370–375, 2009.

[91] Dudley W Reynolds. Repetition in nonnative speaker writing: More than quantity. *Studies in Second Language Acquisition*, 17(2):185–209, 1995.

[92] Filipe N Ribeiro, Matheus Araújo, Pollyanna Gonçalves, Marcos André Gonçalves, and Fabrício Benevenuto. Sentibench-a benchmark comparison of

state-of-the-practice sentiment analysis methods. *EPJ Data Science*, 5(1):23, 2016.

[93] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

[94] Kyle Robinson and Vijay Mago. Birds of prey: identifying lexical irregularities in spam on twitter. *Wireless Networks*, pages 1–8, 2018.

[95] Nicholas Rosso and Philippe Giabbanelli. Accurately inferring compliance to five major food guidelines through simplified surveys: Applying data mining to the uk national diet and nutrition survey. *JMIR Public Health Surveill*, 4(2):e56, May 2018.

[96] H Rubenstein and JB Goodenough. Contextual correlates of synonymy. commun, 1965.

[97] Matthew T Schultz. The intellimetric automated essay scoring engine-a review and an application to chinese essay scoring. *Handbook of automated essay scoring: Current applications and future directions*, pages 89–98, 2013.

[98] Mark D Shermis and Jill Burstein. *Handbook of automated essay evaluation: Current applications and new directions*. Routledge, 2013.

[99] Mark D Shermis and Jill C Burstein. *Automated essay scoring: A cross-disciplinary perspective*. Routledge, 2003.

[100] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[101] Swapna Somasundaran, Brian Riordan, Binod Gyawali, and Su-Youn Yoon. Evaluating argumentative and narrative essays using graphs. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1568–1578, 2016.

[102] Swapna Somasundaran and Janyce Wiebe. Recognizing stances in online debates. August 2009.

[103] César R Souza. Kernel functions for machine learning applications. *Creative Commons Attribution-Noncommercial-Share Alike*, 3:29, 2010.

[104] Christian Stab and Iryna Gurevych. Identifying argumentative discourse structures in persuasive essays. October 2014.

[105] Siti Hamin Stapa and Mohd Mustafa Izahar. Analysis of errors in subject-verb agreement among malaysian esl learners. *3L: Language, Linguistics, Literature®*, 16(1), 2010.

[106] Sandra Stotsky. The vocabulary of essay writing: Can it be taught? *College composition and communication*, 32(3):317–326, 1981.

[107] Tony Stubblebine. *Regular Expression Pocket Reference: Regular Expressions for Perl, Ruby, PHP, Python, C, Java and. NET*. " O'Reilly Media, Inc.", 2007.

[108] Naoko Taguchi, William Crawford, and Danielle Zawodny Wetzel. What linguistic features are indicative of writing quality? a case of argumentative essays in a college composition program. *Tesol Quarterly*, 47(2):420–430, 2013.

[109] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

[110] Yi Tay, Minh C Phan, Luu Anh Tuan, et al. Skipflow: incorporating neural coherence features for end-to-end automatic text scoring. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[111] SM Vohra and JB Teraiya. A comparative study of sentiment analysis techniques. *Journal JIKRCE*, 2(2):313–317, 2013.

[112] Kaja Zupanc and Zoran Bosnić. Automated essay evaluation with semantic analysis. *Knowledge-Based Systems*, 120:118–132, 2017.