SMCC Higher Education Research Journal
ISSN Print: 2449-4402 · ISSN Online: 2467-6322
Volume 5 · August 2018

Essay Scoring System Using Semantic Similarity Approach

DAISA S. OCHARON
http://orcid.org/0000-0002-1439-9070
dsocharon.23@gmail.com
Dean, College of Computer Studies
Saint Michael College of Caraga
Atupan St., Nasipit, Agusan del Norte, Philippines

Gunning Fog Index: 13.13 • Originality: 99% • Grammar Check: 99%
Flesch Reading Ease: 40.91 • Plagiarism: 1%

ABSTRACT

Essay Scoring System provides a systematic checking of essay based on the similarity of its meaning to the model as implied on the content. This study automatically score essays and give feedback to the students about their score. Besides, the system uses algorithms that analyse the lexical semantics of the words to get the similarity between the model and student essay which includes the Common Term Frequency (CTF), Longest Common Subsequence (LCS), and Semantic Distance (SD). The two essay will undergo the Text Processing Phase which includes the process of Tokenization, Stop words removal, Stemming and Parts-of-Speech (POS) tagging. It uses the WorldNet database for word synonymy and semantic references. Word Sense Disambiguation is also implemented in the study to identify the meaning of the word used in the context and also to solve the ambiguity of meaning particularly on homonyms. The scoring follows the predefined criteria for content relevance, spelling, and grammar. Furthermore, the study conducted tests to the actual users of the system including teachers. Based on these tests, the computed percentage differences between the teacher and the system score is 18.03% with an accuracy of 82.18%. The accuracy shows a close similarity of the system's score to the teachers' given score to the essays. Developing the system faces a challenge in the implementation of the semantic algorithms. Since the

study is more capable of evaluating semantic similarity based on word occurrences, it is best to further the system's capability of checking the semantic similarity based on the context of the essay.

KEYWORDS

Natural Language Processing, Computer Science, Information Technology, Essay, Essay Scoring, Semantics, Semantic Similarity, Word Sense Disambiguation, Semantic Distance, Longest Common Subsequence, Common Term Frequency, Tokenization, Stop words, Stemming, WorldNet, Hyponyms, Hypernyms, Asia, Philippines

INTRODUCTION

Scoring of essay exam takes considerably more time especially for large classes and may result from inconsistency across student responses. Scores might be calibrated and analyzed with subjectivity depending on the several interacting factors that may influence the teacher during an assessment.

Numerous researchers assert that the subjective way of essay evaluation leads to variation in grades awarded by different human assessors, which can be analyzed by students as the main root of unfairness (Ade-Ibijola, Wakama, & Amadi, 2012). Computerized scoring can give potential solutions to some of the common shortcomings or issues concerning human essay scoring. Through computer-based scoring of essays which involve the significant accumulation of quantifiable content elements with a specific end goal to assess the nature of an essay can help assessors to ease the process of checking (Zhang, 2013).

Several studies about essay scoring have been conducted to contribute breakthrough, particularly in education. Most of these are using semantic similarity methods to assess the context similarity of essays. In the study of automatic essay grading by Omran and Aziz, it utilizes three algorithms combined in matching phase namely Common Words, Longest Common Subsequence, and Semantic Distance to yield a more efficient result. It uses only synonyms to generate alternative answers of the essay Omran & Ab Aziz, 2013). However, it does not emphasize the meaning of the essay since the computation for semantic distance is not identifying the meaning but based only on the counting of characters and words occurrences that exist in both essays. Further, it does not identify the sense of the word as it is used in the essay taking consideration of how it is used in the context. Besides, it does not integrate the criteria for spelling and grammar checking in scoring.

To address such gaps, the study aims to develop an essay scoring system using semantic similarity approaches which automate the checking of classroom essays and will eliminate subjectivity upon scoring influenced by human scoring. Compare to other research; this study will follow predefined criteria for grammar and spelling

correctness to check for the structure of the essay as well as the relevance of the content to the topic asked. Besides, it will implement hybrid algorithms for semantic similarity which include word sense disambiguation to identify the sense of every word as used in the essay, semantic distance which utilizes the Wordnet to extract similar words for computations, synonym extraction for generated model essay, antonyms to replace word with negative phrases to retain the consistency in meaning, longest common subsequence and common term frequency to give a score more comparable to human.

FRAMEWORK

Text Processing Algorithms

Text processing includes tokenization, stop words removal, stemming and Parts-of-speech tagging. The following shows the algorithm processes.

In this process of tokenization, text sequences are chopped and converted into tokens. The process removes the whitespaces and other punctuations such as {([\t{}():;.])} from the text. Tokens are used for processing string serve as an instance to a particular sequence of characters. These tokens are considered as the semantic unit for processing being grouped. Stop words are most frequently words occur in text and speech. These terms are generic that they have less importance in their meaning. It includes auxiliary verbs and prepositions such as if, as, the, to, at, an, a, what, where, that, on, of and much more (Slimani, 2013; Hussain, 2012). During the stemming process, words will be converted to its canonical word removing all prefixes and suffixes (Jivani, (2011; Jenkins & Smith, 2005). For example, the words nation, nations, nationality and national will have a stem word nation, which determines the similarity of same words with similar meaning but constructed differently using prefixes or suffixes (Moral, de Antonio, Imbert, & Ramírez, 2014). Part-of-speech (POS) tagging is concerned with analysing text and assigning different grammatical roles to each entity (Toutanova, Klein, Manning, & Singer, 2003). It is used to assign grammatical roles as parts of speech such as nouns, verbs, adjectives, adverbs, etc.

Semantic Similarity Algorithms

Semantic similarity algorithms include the Word Sense Disambiguation, Synonym Extractions, Common Words, Longest Common Subsequence, and Semantic distance.

Word Sense Disambiguation

It is a process which selects the appropriate meaning (sense) to a given the word in a text where this meaning is distinguishable from other senses potentially attributable to that word (Kolte & Bhirud, 2009; Bakx, Villodre, & Claramunt, 2006). These senses are the target labels of a classification problem. This algorithm distinguishes and removes the ambiguity of the words used in a concept (Dong, Wang, & Liang, 2015). Using WordNet, words are classified with glosses that identify the senses of a word

according to its parts of speech resulted from POS tagging. The gloss that is, a textual definition of the synset possibly with a set of usage examples (e.g., the gloss of car1 n is "a 4-wheeled vehicle; uses an engine for internal combustion; 'he needs a car to get to work' "). It governs the process of identifying the sense of a word (i.e., meaning) as used in a sentence especially when the word has multiple meanings (polysemy) (Banea & Mihalcea, 2011).

Synonym Extractions

Synonyms are words which are related in meaning (Henriksson et al., 2012). The extraction of synonyms is one of the possible applications in areas of Semantic Web. This is used for query and ontology matching wherein it can be useful in getting the words meaning. Synonym extractions first approach for retrieval is using the dictionaries such as WordNet and Wiktionary (Pearce, 2001). WordNet is a well-established English lexical database that provides meaning and synonyms of a term in different contexts (Montoyo, Suárez, Rigau, & Palomar, 2005). The structure of WordNet is mainly based on the synonym relationship between words (Sun, Huang, & Liu, 2011). These synonyms are grouped into sets called Synsets formed by words that (i) have the same meaning, and (ii) are interchangeable in different contexts. Presently, WordNet contains more than 110,000 Synsets (Lombardi & Marani, (2015).

Common Words (COW)

The algorithm counts the matching words in both answers (the student answer and the generated model answer) where the algorithm works word by word, to determine the number of words that exist in both essays (Omran & Ab Aziz, 2013).

Longest Common Subsequence (LCS)

The algorithm finds the consecutive longest common subsequence of words over the model answers and the student answer. It measures the overlapping consecutive characters presents in a two string. There is the fact that a phrase that is an N-consecutive word is less occurring than an N-non-consecutive word (Macula, Schliep, Bishop, & Renz, 2008). In other words, the longest ones are used less while the shortest is used more. Given an example sequences below for X, Y, and Z:

X: "A B C D E F G H" , Y: "A B C D U V Y K",Z: "A I B T C O D L"

The LCS result is same in both cases (X and Y, X and Z): LCS is "A B C D" and score is the LCS length of 4. It does not differentiate the consecutive relation of their sequence. In this case, Y would be a better choice than Z because it has shared a more consecutive sequence with X. String C is the longest common subsequence (abbreviated LCS) of string A and B if C is a common subsequence of A and B of maximal length,

i.e., there is no common subsequence of A and B that has greater length (Hirschberg, 1977).

Semantic Distance (SD)

Semantic distance is an algorithm used to measure how close or distant the meaning of two units of language is (Budanitsky & Hirst, 2001). The units of language may refer to be phrases, words, sentences, paragraphs, or documents. For example, the nouns dance and choreography are closer regarding their meaning than the nouns clown and building. These units of language, particularly words, may have more than one possible meaning. However, their context may be used to determine the intended senses. For example, a star can mean both a celestial body and a CELEBRITY; however, star, as used in the sentence below, will refer only to the celestial body and is much closer to the sun than to famous: (1) Stars are powered by nuclear fusion to shine brightly in the sky at night.

Semantic distance is of two kind's namely semantic similarity and semantic relatedness. Semantic similarity is a subset or an instance of semantic relatedness, but it can be used differently in a certain context (Budanitsky & Hirst, 2001). Two concepts are determined to be semantically similar if there is hyponymy (hypernymy), antonymy, or troponymy relation between them (Mohammad, 2008).

WordNet

WordNet is a large lexical database that contains the words of the English language. It resembles the characteristics of a thesaurus in that it structures different words that have a similar meaning. It also specifies connections for each of the senses of a given the word (Pedersen, Patwardhan, & Michelizzi, 2004). These connections place words that are semantically related to one another in a semantic network. WordNet is like a dictionary since it describes the definition of words and the corresponding part-of-speech (Miller, 1995).

Synonym relation grouped the primary connection between words, which means that words which are conceptually equivalent, and thus interchangeable in most contexts. These groupings are called synsets which consist of a definition and relations to the other synset (Miller, 1995). A word can be a part of more than one synset, depending on the meaning it bears since it can take more than one meaning. WordNet has a total of 117 000 synsets, which are linked together.

Hypernyms and Hyponyms

Using the Wordnet database, words which are classified as synsets are retrieved and compared to compute the similarity regarding their meaning as used in the essay. Each synset has a set of glosses that defines the meaning of each sense in the repository. To measure the semantic similarity between the two synsets, we use hyponym/hypernym (or is-a relations). Hypernym relations refers to the generic term used to designate a class

of specific instances A is a hypernym of B if B is a (kind of) A. In other words, these are synsets that are more general just like for example the word 'vehicle' will be a hypernym of the car or auto since vehicle is considered as the parent of the two synsets. Hyponyms, on the other hand, are the specific term used to designate a specific member of a class. A is considered as a hyponym of B if A is a (kind of) B. These are synsets that are more specific just like car and auto are hyponyms of a vehicle. Hyponyms determined an "is-a" relationship to their hypernyms (Miller, 1995).

Each synset in Wordnet has a connecting hypernym and hyponyms that has semantic relations that will form a tree structure. This tree structure consists of connecting nodes represented by synsets. The hypernyms will serve as the parent nodes while the hyponyms will form into the child nodes of the hypernyms. Given a tree structure below shows the example of hypernym relations as retrieved from Wordnet (Ferlež, & Gams, (2004).

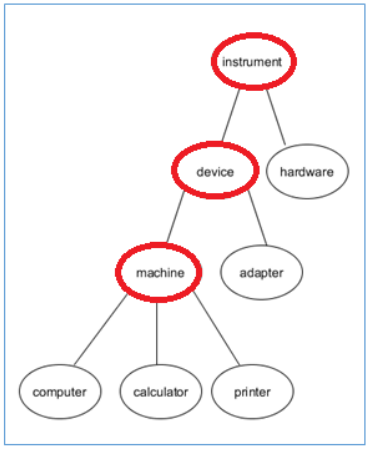

Figure 1: Hypernym of computer

Figure 1 shows the graphical representation of how the hypernym relations in Wordnet is organized. The highlighted nodes are the hypernyms of the synset *computer*. The synset *machine* is the parent of *computer*, *calculator*, and *printer* while the *device* is also the hypernym of *machine* and *instrument* to the *device*. Every hypernym is considered as a parent node to every other synset.

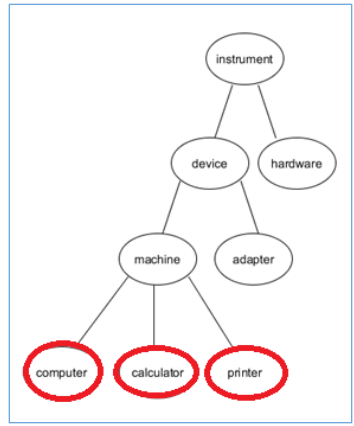

Figure 2: Hyponyms of computer

Figure 2 shows the graphical representation of hyponyms for the synset *machine* which includes the synsets *computer, calculator* and *printer* as shown in the highlighted nodes. These hyponyms will serve as the child nodes to a hypernym, and in the first level, *calculator* and *printer* are the siblings of the computer. While *machine* and *adapter* are hyponyms of *device* as to *device* and *hardware* are also the hyponyms of the *instrument*.

Path-Based Measure of Similarity

Path-based measures are a way of measuring the similarity of text which is calculated as the length of the shortest path between two concepts that connects the concepts through their least common subsumer (LCS). The LCS is the most specific ancestor shared by two concepts. The length is calculated by counting the number of nodes between the two concepts. The lcs-path measure is a modification of this and is calculated as the reciprocal of the length of the shortest path which is shown in the equation below (McInnes, Pedersen, Liu, Melton, & Pakhomov, 2014).

$$sim_{path}(c1, c2) = \frac{1}{minpath(c1, c2)}$$

Where *c1* and *c2* are concepts being paired and is the minimum path between the two concepts through node count.

Node Counting

This is the method of counting the number of nodes between two concepts or words in the tree structure from hypernym and hyponym connections. The count of the node will start from the node of c1 or the first word incrementing by 1 until the node

of the c2 or the second word following the edges that connect those (Ferlež & Gams, 2004). The example tree below shows how the node counting is performed.

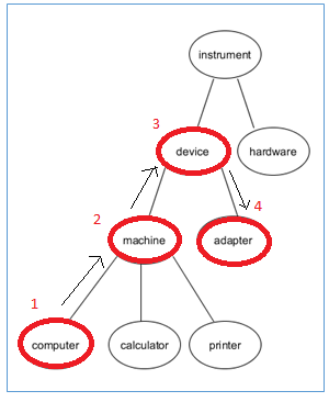

Figure 3: Node counting from computer to adapter

Figure 3 shows the process of counting nodes from computer to the adapter. It shows that synset *computer* to *adapter* has a node count of 4. The count started from the word *computer* following the edge that connects to its parent node and iterating to find the second word.

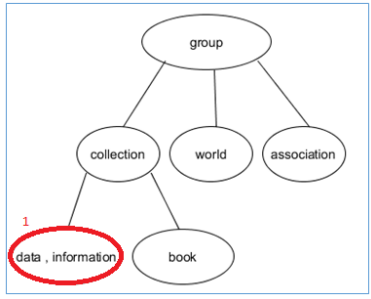

Figure 4: Node counting for synsets data and information

Figure 4 shows another instance of node counting in which both words belong to the same synset, for this case both will be on the same node. Words of the same synset or node imply synonymy which means that the two words have similarity in meaning. Therefore, the node count will be equal to 1 because they belong only to the same node.

OBJECTIVES OF THE STUDY

This study aims to build an essay scoring system that allows students to take an essay-type examination and automatically provide the score. Specifically, it aims to:

1. Perform text processing methods to prepare a data set.
2. Formulate a scoring mechanism for essay based on predefined criteria.

METHODOLOGY

Model and Student Essay

The model and student essay are the primary data inputted to the scoring system where scores are mainly computed from. Model essay refers to the predefined answer set and inputted by the teacher during the adding of a new essay question. It is where the accuracy of the student answer will be based. On the other hand, student essay refers to the student answer or response to the essay questions. It is inputted by the student while taking the essay examination.

Text Processing Methods

Text processing involves the process of tokenization, stop words removal, stemming and Parts-of-speech tagging. Both model and student essays undergo the above processes to prepare the data for semantic similarity methods.

The inputted model and student essay as shown below will undergo the text processing methods as discussed below.

Student Essay	Model Essay
A/DT machine/NN used/VBD to/TO calculate/VB and/CC prcess/NN data/NN.	*A/DT computer/NN is/VBZ used/VBD to/TO give/VB information/NN.*

POS Tagging

Student essay will undergo POS tagging for grammar checking to identify the parts of speech for the words in the essay while model essay uses this method as to prepare for Word Sense Disambiguation process. The example below shows the POS tagging for model and student essay.

The process uses the Brill Tagger class to assign the appropriate parts of speech of the word identified.

Tokenization

Both model and student essay will undergo the process of tokenization. This is done to remove those unnecessary symbols which do not contribute to the meaning of the essay. Table 11 shows the tokenization process for both model and student essay.

Stop Words Removal

After tokenization, stop words are identified and removed to retain only those important words needs during comparison for semantic similarity. Both model and student essay will undergo this process.

During this phase, stop words are identified and removed from the student and model essay. Stop words are identified by comparing the word to the list of stop words stored in the database. After identification, it will then removes and retain the important words. Given the above example, student essay has stop words: *a, used, to, and* while model essay are; *is, a, used,* and *to.*

Stemming

This process is useful in matching words syntactically because it will result in higher similarity count due to its simplified form. The result of the stemming process for student and model essay where the word *computer* is stemmed into *computer* and *information* into *inform*.

Alternative Model Essays

The model essay is the predefined answer by the teacher where the students answer be compared for scoring. It refers to the base idea or thought of the essay topic given by the teacher. Every student essay must be closer to the invoked sense or meaning of the model essay to get a higher score. Comparing student essay to model essay alone set by the teacher would be the direct way of getting the semantic similarity between the two essays. However, the process of analysis would be limited only to the exact words used in both essays. Thus, generating alternative model essays extracted from synonyms or antonyms will give an additional measure of similarity because it will maximize the possibility of using synonymous words in the essay.

Generating alternative model essays is done upon adding a new essay question. Model essay will undergo text processing methods to prepare the data set. Moreover, it will undergo the process of disambiguation using Word Sense Disambiguation (WSD) before getting the synonyms of every word as alternative model essays.

Word Sense Disambiguation (WSD) Process

This is the process of disambiguating every word in the model essay to get the sense of this word as it is used in the context. This is done prior to the generation of the alternative model essay to get the sense of every word used in the essay.

Given the example, the word *computer* stemmed as to *compute* the following glosses or senses:

1. *the procedure of calculating; determining something by mathematical or logical methods*
2. *a machine for performing calculations automatically*
3. *problem-solving that involves numbers or quantities*

4. *the branch of engineering science that studies (with the aid of computers) computable processes and structures*
5. *an expert at calculation (or at operating calculating machines)*
6. *make a mathematical calculation or computation*
7. *may be computed or estimated; "a calculable risk"; "computable odds"; "estimable assets"*
8. *of or involving computation or computers; "computational linguistics."*

Given the above senses, the algorithm will perform comparisons to all the glosses of all the words, the highest count of overlapping words will determine the final sense of the word. Each sense are compared to count the word overlapped as shown on the following equation:

$$Score_{gloss} = \sum_{i=1}^{n} Word_{overlapped}(Gloss_{target\_synset}, \cup Gloss_{synsets-target\_synset})$$

where $Gloss_{target\_synset}$ refers to the gloss of the synset being compared while $\cup Gloss_{synsets-target\_synset}$ is the union of all the glosses of other synsets except for the target synset ready for comparison. The refers to the function that will count the overlapping or matching words during the comparison of glosses. This function will continue until all words per glosses will be compared. To get for the sense of each word, the following equation is used:

$$Sense_{word} = Max(Score_{gloss_1}, Score_{gloss_2}, Score_{gloss_n})$$

where the maximum value of the score per glosses will be extracted as the final sense of the target word.

Each gloss will be tokenized to compare each word to other glosses. Comparison of all the words in the essay will continue until all words are compared and counted the overlapping words occur in each gloss. As the loop ends, the program will get the gloss with the highest value of counts and declared it as the final sense of the word.

From the above senses, as cited in the example of *computer*, the final sense of the word *computer* is "*a machine for performing calculations automatically*."

Synonym and Antonym Extraction

Synonyms are extracted from the WordNet synsets after the appropriate sense of the word is identified. The set of synonyms are classified according to the sense of every word in which it is linked to its corresponding synonymous terms in database. Based on the synonyms above in every word, it will generate an alternative essay such as *computer, distinguished, invention, modern* and *engineering.* Another alternative model essay would be *computing machine, eminent, innovation, modern* and *engineering science* and so on.

Additionally, antonyms are also extracted if there are negative phrases found before or after the word such as *not, no, don't, never, nothing* and so on and remove the negating word which is also considered as stop words which normally be removed during text processing. However, before removing this as stop words, after tokenization process words with negative phrases are identified and replaced by its corresponding antonyms instead of synonyms to maintain the consistency of the meaning of the phrase or context. Example the phrase *not beautiful* may similar to *ugly* than to *beautiful* when *not* is removed during stop words removal.

Given the synonyms or antonyms extracted for every word in the model essay, it will replace the original word and will generate another model essays as an alternative.

Criteria and Algorithm Weights

Weights are necessary for the study because it maintains the balance of importance between the resulted scores of the algorithms. It helps to level priorities accordingly in scoring wherein it takes a higher weight to the algorithm result which is believed to contribute higher value in the final score. Scores are computed based on the standard weights defined for each criterion in essay checking.

The researcher has set defined weight distribution for the different criteria since no study that cited the appropriate weight distribution for similarity algorithms and the criteria for grammar and spelling. Semantic similarity algorithms have a weight distribution of 20% for CTF, 20% for LCS and 60% for SD. This weight distribution offers the best balance for the three algorithms wherein SD has an above 50% compares to the other algorithms. These given weights are customizable in the administrator account for the possibility of updates. Moreover, the weight distribution for spelling and grammar is both 15% to give the best balance of importance for both grammaticality and spelling. This weight is set as the maximum, the teacher could still customize the weight within the given range. These weights are based on the presumptions taken from the experts and the formulated weights of the researcher and are experimented using the scoring system.

Semantic Similarity Algorithms

Algorithms for measuring semantic relations include Common Term Frequency (CTF), Longest Common Subsequence (LCS) and Semantic Distance (SD). The mentioned algorithms have its corresponding weight in computing similarity. The result of the computation will be the score for content relevance criteria of the essay.

Common Term Frequency

The algorithm checks the matching words from the student essay with the model and the generated model answers. It will only count the matching non-repeating words for every essay being compared. It works by comparing the student essay to model essay and counting the words common to both essays. These words must be the same string.

The repetition of these strings in the essay is counted only once. Given is an example below:

Student Essay: Computer is used to calculate numbers.
Model Essay: Machines make work easier. Computers are considered machines. This machine computer is used to calculate.

The above example shows that in the model essay, the word *computer* matches with the student essay and it appears twice thus it is counted only once to avoid a higher score for answers with repeating words. Another word *calculate* match with student essay with a total count of *2* over the total number of words in the model essay. The possibility of matching exact words between essays is low and it is rare to happen to the majority of word occurrences. Common Term frequency will be computed as follows:

$$\boldsymbol{CTF_{essay}} = \frac{\boldsymbol{match_{word_{count}}}}{\boldsymbol{total_{words_{model}}}}$$

where $match_{word_{count}}$ refers to the number of distinct words that matches between two essays and $total_{words_{model}}$ is the total number of words present in the model essay that is used during the comparison.

The matching will continue to all generated model essays and the result with the highest CTF value is considered as the final CTF.

Longest Common Subsequence

The algorithm identifies the longest common substring or the consecutive characters that match between student essay with the model essay and generated models. The count is not limited to a single word only but could be extended as a whole sentence or even a paragraph. It will get the exact matching string between the two essays. Example, the given the student and model essay respectively.

Student Essay: Nature is God's creations and gift to everyone.
Model Essay: God's creation is a gift to everyone.

Based on the above example, the longest common substring between the two essays is *God's creation* because it overlaps in both essays. Although there are other words that matches but the algorithm selects the longest substring. Thus, the count for the longest substring is 12 ignoring the apostrophe and whitespace which can be removed during the tokenization process. The matching will continue until all generated models are matched to student essays. The LCS with the highest value will be the final LCS. Longest Common Subsequence is computed as follows:

$$\boldsymbol{LCS_{essay}} = \frac{\boldsymbol{longest_{characters_{count}}}}{\boldsymbol{total_{characters_{model}}}}$$

Where $longest_{characters_{count}}$ refers to the character count of the longest matching substring between two essays while $total_{characters_{model}}$ refers to the total number of characters present in the model essay.

Semantic Distance

The algorithm measures how the two essays are similar or distant regarding their meaning despite having different words used in the essay. Semantic distance holds between lexical items having a similar meaning. This will emphasize the concept of the semantic similarity between words and the essay in general.

It compares each word in the model essay to every word in the student essay forming into pairs and getting the computed semantic distance of that pairs. Computation of the semantic distance per word pairs is the prior process before calculating the similarity of the two essays in general. How the computation process is done will be discussed thoroughly in the next paragraphs. Thus, the semantic distance is computed in two phases, semantic distance between words and between essays.

A.***Semantic Distance between Words***

This is the prior process in the computation of the semantic distance between essays because the resulted semantic distance between words will be the basis upon comparing each word in the essay. In this process, the pair of words that are semantically similar is defined with its corresponding distance value that determines how the two words are similar in terms of their meaning. This algorithm utilizes the words in the Wordnet database wherein each word will be paired with its similar words based on the hypernym and hyponym relations.

Each pair will be measured in its semantic distance depending on the count of the connecting nodes between them. This is called the path similarity, and it is equal to . It ranges from 0.0 (least similar) to 1.0 (identical).

The computation of the semantic distance between words is achieved by implementing the following important processes or steps:

1. ***Querying hypernyms and hyponyms from Wordnet***

This is the process of extracting the hypernyms and the hyponyms of every word which identifies similarity to the target word. These two relations are important in identifying the similarity of words in terms of their connections as presented in Wordnet. Each relation is stored in separate tables in Wordnet. Every synset is linked to its corresponding hypernyms and hyponyms as represented by its synset_id which can be formed into a tree-like structure linking node of concepts to other concepts. Thus, Figure 36 shows the snippet query of getting the values for hypernyms of the target word.

The query will get all the hypernyms which represent the parents of the target word forming into hierarchy ending to the root node of the tree which is the last hypernym value. Using the word *computer* as an example, the query will retrieve the *machine* as the parent node of the *computer*, while the *device* is also the parent of *machine*, and *instrument* is the parent of the *device*. This result shows a relation between words such as the *computer* is a kind of *machine* as well as the *machine* is a *device* and so on. On the other hand, the hyponyms of each hypernym will be queried as follows from the hyponym table in Wordnet. The resulted hyponyms served as the children of each hypernym. Every hyponym in every level of the tree hierarchy is considered siblings.

Given the example, as discussed previously, hyponyms retrieve for the synset *machine* and at same time siblings of the *computer* are the synset *calculator* and *printer* as part of it. For the synset *device*, it retrieved hyponyms *machine* and *adapter* while the *instrument* is *device* and *hardware* respectively.

The more specific processes in querying these relations will be discussed thoroughly in the next process.

2. ***Identifying and Linking Nodes***

This is the process of identifying the hypernyms and hyponyms to be linked to a given word which can be the linking nodes in the tree structure. These relations are connected based on its given synset_id per synset/word. It is the reference value to each relation that links to every other synset. Identifying the synsets that links to a specific word in the tree hierarchy is a bit tricky and requires a focus tracing. This process will utilize the three important relations in Wordnet namely *synsets*, *hypernyms* and *hyponyms*. To understand the whole process fully, the following sub-processes are executed.

2.1 ***Querying the target word***

This process needs to query the target word to get the available senses of that word which serves as the starting point for all processes. Every sense of the word will determine how the word will be used or mean. The resulting synset_id from the query has its corresponding linked hypernyms and hyponyms that will be used later in the next processes.

The synset *computer* has two senses represented by the synset_id *102971359* and *109257296*. These synset_d will be the starting reference value in querying the hypernyms and hyponyms of the target word iterating until the root node will be identified. Each of the senses has a different link of synsets that can be formed into a different tree hierarchy.

2.2. ***Querying the hypernyms of the target word***

This process will retrieve the hypernyms that link in each sense of the target word. To showthe process, it uses the sense *102971359* as an example based on the previous query.

The column *synset_id_2* represents the hypernym of the synset_id in the first column. Since synset_id *103561924* is the first resulting hypernym of the query, it is considered as the parent node of the synset *102971359*. Since the hypernym values shows only the synset_id, by querying the value from synset table as what is done from the previous process, it shows that synset id *103561924* is *machine*, thus the hypernym of *computer* is *machine* as shown in the resultset From this point, the process will still iterate continuing the query from the synset_id *103561924* to get also its parent node until it reaches the last resulting hypernym value which is considered as the root node of the tree.

2.3 ***Querying the hyponyms of the target word***

This process will retrieve the linking hyponyms in each sense of the target word. The resulting hyponyms will serve as the child nodes of the hypernym as queried from the previous process. Upon the hypernym is identified, it will automatically find all the hyponyms linking to it as its child nodes in the tree structure.

To check for the hyponyms of the synset *machine*, it is queried in the hyponym table as shown in the resultset. It had 42 hyponyms and shown as highlighted in the word *computer* which means that it is one of the hyponyms as expected. These hyponyms will be the child nodes of synset *machine* in the tree hierarchy, and the first level serves as the siblings of the synset *computer*.

After understanding the above discussions, the tree structure will partially be in the first hierarchy. The tree structure shows only some of the hyponyms for graphical representation, but the actual resultset is 42 rows with *computer* 102971359, *ATM* 102870954, *calculator* 102834550, *decoder* 103054310, *printer* 103853110 and *motor* 103647935 as part of it.

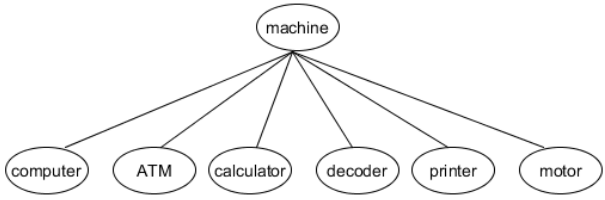

Figure 5: Tree structure showing the first level hierarchy

To add the next higher level, the algorithm must know again the hypernym of the *machine* through its synset_id to connect as the parent node through following the repetitive processes discussed above.

This is the continuation from above process wherein it shows that synset *103561924* has a hypernym or parent node of synset *103068033* which is referring to synset *device*

with 119 fetched hyponyms including the synset *machine* as expected. The same repetitive process will be done until the root node is identified.

3. ***Counting the Path Node***

The process will count the number of nodes linking the path between the two words. This count will be used in the formula for getting the semantic distance. The synset *computer* to *adapter* has a node count of 4. The count started from the word *computer* and it follows three edges that connect four nodes including the word *adapter* passing through the parent of *computer*. Since the *machine* is the parent node of the *computer*, then the node count between the two synsets is 2 because there is only one edge that connects them.

The synset *computer* and *printer* are both hyponyms of *machine*. Therefore, both are of the same parent and level. Thus, the node count between the two synsets is 3 because starting from the word *computer*, it passes one node which is the parent node *machine* going down to the edge that connects *machine*, and *printer* since the *printer* is a child node of *machine*. Therefore, it passes only 3 nodes which include a *computer, machine* and *printer*.

Since synset *computer* and *calculator* are both hyponyms of machines and like a *printer*, both shares the same parent. Therefore, there are only two edges that connect the two synsets with their parent which also yield to node count of 3. The synset *computer* and *instrument* has a node count of 4 because there are 4 connecting nodes starting from the node *computer,* passing through *machine* and *device* until to the node *instrument* that passes along the hypernyms path of the computer.

Besides, synset *data* and *information* resulted from the query are belong to the same node. Therefore, the node count will be equal to 1 because no edge that separates them rather the words are in the same node. This instance will happen when a certain synset will retrieve more than one word during the query like for the given example below. The words with the same synset_id belong to the same synset which also means that they are synonyms and both in the same node in the tree structure. The words *data* and *information* has both synset_id of *107949563* that identifies them in the same node.

4. ***Computation of Semantic Distance***

This process will compute the semantic distance between a pair of words based on the count of nodes resulted from the previous process. It will compute the semantic similarity of two words using the path similarity measures which is represented by the equation below.

$$\boldsymbol{SD(synset1, synset2)} = \frac{\mathbf{1}}{\boldsymbol{node}_{\boldsymbol{distance(synset1, synset2)}}}$$

Where is the path length from *synset1* to *synset2* using node counting. Based on the example, the semantic distance for synset *computer* and *machine* is ½ or 0.5 similarity.

On the other hand, the word *computer* and *calculator* have a count of 3 nodes which also yield to 1/3 or 0.33 similarities. The synset *computer* and *instrument* has a node count of 4 which also results to ¼ or 0.25. The synset *calculates* and *computes* are of the same synset thus the similarity between the two is 1/1 or 1. The similarity for *data* and *information* is also 1/1 or 1 since the two are of the same synset. Lastly, the synset *gave* and *calculate* node count of 3 which yields to 1/3 or 0.33 similarities.

5.***Generating and Storing Semantic Distance of Word Pairs***

This is the process of deployment which is necessary before the computation of the semantic distance for the whole essay. It is where the word pair semantic distance value is retrieved during the comparisons for two essays. It will generate all the computed semantic distance of word pairs based on the above computations discussed from previous processes and stores this in a separate table in the database that serves now as the reference or lookup for other computations. As a result of deployment of the process, the table now contains more than 5 million of similar words in pairs with a corresponding semantic distance computation in each pair.

This collection will be used in the computation of semantic distance in the essay as the final process.

B.***Semantic Distance between Essays***

After generating computed semantic distance between word pairs as discussed from previous sections, the computation of similarity between essays will follow. This process will use the stored calculated distances between word pairs in the comparison of the two essays. Each word in the model essay will be compared to every word in the student essay forming into a pair of words to get the average similarity for the two compared essays. Every pair of words will be compared to the stored pair in the database for the semantic distance value, in the case that the pair does not exist in the database; it means that the two words have no similarity in meaning. The algorithm will compute the average semantic distance of all the word pairs with a semantic distance value taken from the database which is represented by the following equation.

$$\boldsymbol{average}_{distance} = \frac{\boldsymbol{total}_{distance_{pair}}}{\boldsymbol{word}_{pair_{count}}}$$

where $\boldsymbol{total\_distance}_{pair}$ is the summation of all the computed semantic distance in all word pairs and $\boldsymbol{word\_pair}_{count}$ is the total word pairs. To check the percentage occurrence of similar words between essays, the following equation is used

$$\boldsymbol{occurrence}_{percentage} = \frac{\boldsymbol{word}_{count_{model}}}{\boldsymbol{total}_{words_{model}}}$$

where $\boldsymbol{word\_count}_{model}$ refers to the word count with an existing semantic value as compared to the database after pairing to student essay while $\boldsymbol{total\_words}_{model}$ refers to

the total number of words used in the model essay.

The occurrence percentage is considered in the computation to check the number of words in the model essay with a semantic value existing after pairing to the student essay. It is necessary to maintain fairness in scoring. For instance, the model essay has 5 words and upon pairing, there are 1 out of 5 words having a semantic distance value and it is 1 that may result in an average of 1 which is unfair for those with a count of 4 out of 5 with lower semantic distances. Though it is lower in average, it also means that the essay with a higher number is using words more similar to the other that's why it is given weight in computation.

The average between the above percentage and average distance will be the semantic distance between the two essays compared.

Based on the above computations, the semantic distance between the two essays will be computed as follows:

$$\boldsymbol{SD\_essay} = (\boldsymbol{occurrence}_{\boldsymbol{percentage}} + \boldsymbol{average}_{\boldsymbol{distance}}) \boldsymbol{/ 2}$$

Grammar and Spelling Checking

Checking both the grammar and spelling in the student essay plays an important criterion in evaluating the structure, grammaticality, and style of writing.

Spelling Checker

Spelling checks the correctness of form and combination of letters used in a word. It signifies the meaning of the word itself. Thus, misspelled word will change the thought being conveyed in the sentence or the whole essay. Spelling checking is done by comparing each word in the essay to the collection of words in the database. Every word which does not match is considered misspelled. Proper Nouns must start in uppercase. The spell checker is not autocorrected. This student essay must be sensitive to the word being typed. Based on the example student answer below the highlighted word signifies wrong in spelling.

Errors in spelling are counted and be divided into the total words of the essay. The result will then be multiplied to the points allocated for spelling which is equivalent to the percentage set by the teacher upon adding a new essay. The computation is as follows:

$$\boldsymbol{SC} = [\frac{\boldsymbol{total}_{\boldsymbol{words}} - \boldsymbol{error}_{\boldsymbol{count}}}{\boldsymbol{total}_{\boldsymbol{words}}}] * \boldsymbol{spelling}_{\boldsymbol{points}}$$

Where is $spelling_{points}$ the weighted maximum score for spelling? Given an example as identified spelling error which is 1, will be computed as follows:

$$SC = [\frac{(8-1)}{8}] * 2.25 = \mathbf{1.96875}$$

Where **1.96875** is the percentage score of the total score set by the teacher with the percentage weight of the spelling? In this case, the teacher set a total score to 15 and allocating 15% to it for spelling. It is derived by multiplying the score percentage allocated to spelling to the total essay score both set by the teacher which are computed as follows:

$$spelling_{points} = total\_essay_{score} * spelling_{percentage}$$

The result for spelling will be added to the resulted score for grammar and relevance.

Grammar Checker

Without good grammar, clear communication and writing are nearly impossible. Proper grammar keeps essay from being misunderstood while expressing the thoughts and ideas that the essay wants to convey. Grammar has rules that must be followed to be consistent with the thoughts expressed in the student answer. Some of these are the proper use of punctuations, capitalizations and subject-verb agreement and some other special cases like the use of *an* or *a*. The algorithm used for grammar checking followed proper precedence of words as organized and identified according to each part of speech. It uses the process of POS tagging for assigning tags to each word. After tags are assigned, grammar is checked by defining some important rules of precedence of what comes before or after specific parts of speech. Example, preposition like *about, for, in, from* and much more are words used to link nouns, pronouns, or phrases to other words in a sentence. Prepositions are usually short words, and they are normally placed directly in front of nouns. In some cases, you'll find prepositions in front of gerund verbs. Personal pronouns like *she*, *he, they* are a pronoun that is associated primarily with a particular person, in the grammatical sense. These pronouns can usually follow a verb, adjective and placed after any prepositions. The student answer below is shown with POS tags.

Not conforming to these basic rules will count an error for grammar. Grammar checking is computed as follows:

$$\boldsymbol{GC} = [\frac{\boldsymbol{total_{words} - error_{count}}}{\boldsymbol{total_{words}}}] * \boldsymbol{grammar_{points}}$$

Where $total_{words}$ refers to the number of words in the essay while $error_{count}$ is the total errors identified in grammar. Further, $grammar_{percentage}$ is the weighted score resulted from multiplying the percentage set by the teacher for grammar and the total essay score as computed below:

$$grammar_{points} = total_{essay_{score}} * grammar_{percentage}$$

Same with spelling, the resulted score for grammar will be added to spelling and relevance weighted score. In the example of the student essay, it shows that there is no identified error in grammar with $error_{count}$ equals to 0 which is computed as follows:

$$GC = [\frac{8-0}{8}] * 2.25 = \mathbf{2.25}$$

Where **2.25** is the 15% of the total score that is allocated for the perfect score of the essay which is 15 the same case with the spelling.

Final Scoring Computation

Given the results from grammar, spelling and the semantic similarity algorithms, the final score for the essay is computed. Essay scoring follows the predefined criteria for spelling, grammar and content relevance which given weights are discussed from previous sections. Computation for the final score of the essay given the previous examples is discussed below.

Computation for Content Relevance

The computation for content relevance is based on the results in the computation for CTF, LCS, and SD. The highest resulting value from each algorithm will be selected and used in the computation.

To compute for a weighted score for the relevance of the content, the following equation is followed:

$$\boldsymbol{CR} = [(\boldsymbol{Highest_{CTF}} * \boldsymbol{Weight_{CTF}}) + \boldsymbol{Highest_{LCS}} * \boldsymbol{Weight_{LCS}}) + (\boldsymbol{Highest_{SD}} * \boldsymbol{Weight_{SD}})] * \boldsymbol{criteria_{points}}$$

where the highest value for the three semantic similarity algorithms will be multiplied to each standard weights to get the weighted score for each method. The result for this will be multiplied to the $\boldsymbol{criteria_{points}}$ which refer to the highest points allocated to content relevance as the percentage score set by the teacher which is computed as follows.

$$criteria_{points} = total_{score_{essay}} * relevance_{percentage}$$

where the $relevance_{percentage}$ is the deducted percentage from the allocated spelling and grammar percentage set by the teacher to each essay question.

To replace the equation with the given an example above and based on the computed results for CTF, LCS, and SD as discussed from previous sections, it is computed as follows.

$$CR = [(0 * 0.20) + (0.2592 * 0.20) + (0.85375 * 0.60)] * 10.5$$
$$CR = [0 + 0.05184 + 0.51225] * 10.5$$
$$CR = 0.56409 * 10.5$$
$$\boldsymbol{CR = 5.922945}$$

The score for the content relevance of the essay based on the results of the three algorithms is **5.922845**.

Computation of the Essay Score

Given the score distribution of the essay as computed based on the percentage set by the teacher, the score distribution for each criterion based on the given example discussed previously are as follows:

$$\boldsymbol{Highest\_Score_{Essay} = 15}$$
$$\boldsymbol{Grammar_{Score} = 2.25}$$
$$\boldsymbol{Spelling_{Score} = 2.25}$$
$$\boldsymbol{Content\_Relevance_{Score} = 10.5}$$

The teacher will set the $Highest\_Score_{Essay}$ which is the maximum or perfect score for the essay. On the other hand, $the\ Grammar_{Score}$ and $Spelling_{Score}$ is the 15% of the perfect score as what the teacher set to the essay along with the $Content\_Relevance_{Score}$ which is the 70% of the $Highest\_Score_{Essay}$ from the deduction of 30% from grammar and spelling.

Given the computation for spelling and grammar as shown from previous sections, the final essay score is computed as follows:

$$\boldsymbol{Score_{essay} = GC + SC + CR}$$

where *GC* is the computed score for grammar, *SC* for spelling and *CR* for content relevance. The computed score for grammar based on the previous calculations is *2.25*, spelling *SC* is **1.96875** and content relevance *CR* is **5.922845**.

$$\boldsymbol{Score_{essay} = 2.25 + 1.96875 + 5.922945 = 10.141695}$$

The final score of the student essay is **10.141695** out of 15 as the highest possible score. The result shows that the essay scores more than 50% of the total score. The score for content relevance shows a higher score which means that the two essays are more similar in terms of their meaning. The final score signifies that the algorithm can give emphasis on the similarity of meaning despite the differences of the words used in the essay.

RESULTS AND DISCUSSION

Grammar and Spelling Error Evaluation Accuracy Test

The comparisons of grammar and spelling error evaluation accuracy test between the teacher expectation and the actual result generated by the system. The system indicates the POS tags for each word and highlighting the errors evaluated.

The percentage error between the errors counted by the teacher and the system were determined and to calculate the individual accuracy, the resulted percentage difference

will be deducted from 100. The overall accuracy is the average of all the individual accuracy being computed. The computations are shown below to get the percentage error and accuracy. Given the values for spelling 5 and 5, the percentage of difference is 0% meaning there is no difference between the values while the computed accuracy is 100% while grammar with 4 and 3 resulted in percentage difference of 28.57% with accuracy of 71.43% which means that the error counted by the system differs from about 28.57% to teacher counts which is 71.43% accurate to the teachers evaluation.

Based on the above results, the computed overall percentage difference for spelling is 8% with an accuracy of 92% while grammar has 30% percentage error and accurate for about 70%. The overall results show that spelling and grammar checker can identify errors very similar to human.

Computation of Percentage Difference and Accuracy

The percentage difference between the final scores of two essays is computed using the following formula:

$$\boldsymbol{Percentage_{difference} = |\frac{First_{value} - Second_{value}}{(First_{value} + Second_{value})/2}| \, x \, 100\%}$$

Where $First_{value}$ and $Second_{value}$ will refer to the scores of both essays. Replacing the equation by example:

$$\begin{aligned} Percentage_{difference} &= |\frac{25 - 25.77}{(25 + 25.77)/2}| \, x \, 100\% \\ &= |\frac{0.77}{(50.77)/2}| \, x \, 100\% \\ &= |\frac{0.77}{25.385}| \, x \, 100\% \\ &= |0.030| \, x \, 100\% \\ &= \mathbf{3.03\%} \end{aligned}$$

The result for percentage difference signifies the absolute value of how the final score rated by the teacher differ from those of the system. On the other hand, accuracy of system scores to the teachers score is computed as follows:

$$\boldsymbol{Accuracy = 100\% - Percentage_{difference}}$$

Thus, using the above result, the accuracy of the scores is:

$$\begin{aligned} Accuracy &= 100\% - 3.03\% \\ &= \mathbf{96.97\%} \end{aligned}$$

Essay Scoring Accuracy Testing

It summarizes the percentage difference and accuracy of all test cases which results in the overall average of 18.03% difference from the scores given by the teacher

compares to the system score. Besides, the result also shows an accuracy of 82.18% for the compared scores from the teacher and the system.

Table 1. Test Case Results Summary

Test Cases	Percentage Difference			Average Percentage Difference	Accuracy			Average Accuracy
	Student 1	Student 2	Student 3		Student 1	Student 2	Student 3	
Test Case 1	17.58%	24.07%	17.18%	19.61%	82.42%	75.93%	82.82%	81.05%
Test Case 2	24.63%	20.18%	7.11%	17.30%	75.37%	79.82%	92.89%	82.69%
Test Case 3	8.98%	22.32%	20.27%	17.19%	91.02%	77.68%	79.73%	82.81%
Average	18.03%				82.18%			

Performance Testing

It tested the performance of the system in terms of its execution time in executing the processes in scoring. The results may vary based on the inputted characters or number of words that the essay contained. The time of generation of word pair semantic distance during deployment including the computation and saving to the database.

Table 2: Performance Testing for Generating the Semantic Distance of Word Pairs

No. of Word pairs	Execution Time (minutes)
1,009,124	14.14
2,001,637	27.36
3,001,234	42
4,002,548	58.38
5,153,537	81

Additionally, the performance test for the process of computation based on the number of inputted words in the student essay is shown in Table 3.

Table 3: Performance Testing for the Process of Computation of Essay

No of words in Student Essay	Execution Time (minutes)
30	0.13
50	0.14
100	0.20
200	0.30
500	0.48
1000	0.56
2000	0.88
4000	0.91

Based on Table 3, it shows that the execution time depends on the number of words present in the student essay.

CONCLUSIONS

Using Word Sense Disambiguation, word senses as used in the context is identified to get the meaning of the essay. Although the system focuses on the word semantics, the use of synonyms and antonyms also helps to get the hidden meaning of the words used that can contribute to the sense of the whole essay. The system also enables grammar and spelling checking that can help the teacher to assess the correctness of grammar and the spelling of every word used in the essay.

Through the combinations of the algorithms implemented in the system, the teacher can minimize the time and effort in checking the essays especially for classes with a large number of students. Moreover, the software also can provide scores that are not influenced by any factors of subjectivity compared to manual checking. It can handle score consistency and closer similarity to a human checker. Based on the findings taken, the scoring system can provide scores to student essays closely similar to human scoring.

ACKNOWLEDGMENTS

This study has been made possible because of the contributing sources cited and the help and support of different personalities who contributed to the development of this study. To Ms. Marisa Buctuanon, the thesis adviser, who always supervised, encourages and shared her knowledge to the researcher untiringly during the development of the program and the documentation process. To the Dean of CICCT, Dr. Gregg Victor Gabison and the rest of the faculty and staff whose critics and advice challenged me to pursue and to give the best I can do in this study. To my SMCC administrators and colleagues who always supported my endeavor and giving me enough time during the development. To my family who always gives me moral support and inspiration to finish the work especially to my daughter whose hugs and kisses inspire me more to continue what I'm doing. Lastly and above all, to our Almighty God for always giving me strength and wisdom to endure despite the different struggles encountered during the development of this study.

LITERATURE CITED

Ade-Ibijola, A. O., Wakama, I., & Amadi, J. C. (2012). An expert system for automated essay scoring (AES) in computing using shallow NLP techniques for inferencing. *International Journal of Computer Applications*, *51*(10). Retrieved on January 14, 2019 from https://goo.gl/gNq7ee

Bakx, G. E., Villodre, L. M., & Claramunt, G. R. (2006). Machine learning techniques for word sense disambiguation. *Unpublished doctoral dissertation, Universitat Politecnica de Catalunya*. Retrieved on January 14, 2019 from https://goo.gl/UCDdNZ

Banea, C., & Mihalcea, R. (2011, January). Word sense disambiguation with multilingual features. In *Proceedings of the Ninth International Conference on Computational Semantics* (pp. 25-34). Association for Computational Linguistics. Retrieved on January 14, 2019 from https://goo.gl/xpJuqG

Budanitsky, A., & Hirst, G. (2001, June). Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and other lexical resources* (Vol. 2, pp. 2-2). Retrieved on January 14, 2019 from https://goo.gl/ad7xdo

Dong, H., Wang, W., & Liang, H. N. (2015, December). Learning Structured Knowledge from Social Tagging Data: A Critical Review of Methods and Techniques. In *Smart City/SocialCom/SustainCom (SmartCity), 2015 IEEE International Conference on* (pp. 307-314). IEEE. Retrieved on January 14, 2019 from https://goo.gl/4vzjUc

Ferlež, J., & Gams, M. (2004). Shortest-path semantic distance measure in wordnet v2. 0. *Information Society in 2004*, 381. Retrieved on January 14, 2019 from https://goo.gl/i5UFPW

Henriksson, A., Moen, H., Skeppstedt, M., Eklund, A. M., Daudaravicius, V., & Hassel, M. (2012). Synonym extraction of medical terms from clinical text using combinations of word space models. *Proceedings of Semantic Mining in Biomedicine (SMBM). Institute of Computational Linguistics, University of Zurich*, 10-17. Retrieved on January 14, 2019 from https://goo.gl/fiybT2

Hirschberg, D. S. (1977). Algorithms for the longest common subsequence problem. *Journal of the ACM (JACM)*, *24*(4), 664-675. Retrieved on January 14, 2019 from https://goo.gl/aGdUAE

Hussain, A. (2012). Textual Similarity. *Bachelor Thesis. Kongens Lyngby: University of Denmark*. Retrieved on January 14, 2019 from https://goo.gl/3DQAjZ

Jenkins, M. C., & Smith, D. (2005, August). Conservative stemming for search and indexing. In *Proc. ACM SIGIR*. Retrieved on January 14, 2019 from https://goo.gl/puuhdw

Jivani, A. G. (2011). A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, *2*(6), 1930-1938. Retrieved on January 19, 2019 from https://goo.gl/pV31cJ

Kolte, S. G., & Bhirud, S. G. (2009). WordNet: a knowledge source for word sense disambiguation. *International Journal of Recent Trends in Engineering*, *2*(4). Retrieved on January 14, 2019 from https://goo.gl/eSuSXq

Lombardi, M., & Marani, A. (2015, October). SynFinder: a system for domain-based detection of synonyms using WordNet and the web of data. In *Mexican International Conference on Artificial Intelligence* (pp. 15-28). Springer, Cham. Retrieved on January 14, 2019 from https://goo.gl/J7rwjF

Macula, A. J., Schliep, A., Bishop, M. A., & Renz, T. E. (2008). New, improved, and practical k-stem sequence similarity measures for probe design. *Journal of Computational Biology*, *15*(5), 525-534. Retrieved on January 14, 2019 from https://goo.gl/7ZabpM

McInnes, B. T., Pedersen, T., Liu, Y., Melton, G. B., & Pakhomov, S. V. (2014). U-path: An undirected path-based measure of semantic similarity. In *AMIA Annual Symposium Proceedings* (Vol. 2014, p. 882). American Medical Informatics Association. Retrieved on January 14, 2019 from https://goo.gl/AUzCAj

Mihalcea, R., Corley, C., & Strapparava, C. (2006, July). Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI* (Vol. 6, pp. 775-780). Retrieved on January 14, 2019 from https://goo.gl/U16LNp

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39-41. Retrieved on January 14, 2019 from https://goo.gl/QfBCgG

Mohammad, S. (2008). *Measuring semantic distance using distributional profiles of concepts* (Doctoral dissertation). Retrieved on January 14, 2019 from https://goo.gl/1JU8VR

Moral, C., de Antonio, A., Imbert, R., & Ramírez, J. (2014). A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, *19*(1), n1. Retrieved on January 14, 2019 from https://goo.gl/XCNBa9

Montoyo, A., Suárez, A., Rigau, G., & Palomar, M. (2005). Combining knowledge-and corpus-based word-sense-disambiguation methods. *Journal of Artificial Intelligence Research*, *23*, 299-330. Retrieved on January 14, 2019 from https://goo.gl/Gm7Saz

Omran, A. M. B., & Ab Aziz, M. J. (2013). Automatic essay grading system for short answers in English language. *Journal of Computer Science*, *9*(10), 1369. Retrieved on January 14, 2019 from https://goo.gl/teB7rx

Pearce, D. (2001, June). Synonymy in collocation extraction. In *Proceedings of the workshop on WordNet and other lexical resources, second meeting of the North American chapter of the association for computational linguistics* (pp. 41-46). Retrieved on January 14, 2019 from https://goo.gl/LgJ4Gu

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004, May). WordNet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at HLT-NAACL 2004* (pp. 38-41). Association for Computational Linguistics. Retrieved on January 14, 2019 from https://goo.gl/BHD1v2

Slimani, T. (2013). Description and evaluation of semantic similarity measures approaches. *arXiv preprint arXiv:1310.8059*. Retrieved on January 14, 2019 from https://goo.gl/bu6Tev

Sun, K. T., Huang, Y. M., & Liu, M. C. (2011). A WordNet-Based Near-Synonyms and Similar-Looking Word Learning System. *Journal of Educational Technology & Society*, *14*(1). Retrieved on January 14, 2019 from https://goo.gl/H3XpHJ

Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003, May). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1* (pp. 173-180). Association for Computational Linguistics. Retrieved on January 14, 2019 from https://goo.gl/AtnF7t

Zhang, M. (2013). Contrasting automated and human scoring of essays. *R & D Connections*, *21*(2). Retrieved on January 14, 2019 from https://goo.gl/vzVmSJ