Exploiting semantic similarity models to automate transfer credit assessment in academic mobility

by

Dhivya Chandrasekaran

A thesis submitted in partial fulfillment of the

requirements for the degree of

Master of Science

in

Computer Science

in the

Faculty of Science and Environmental Studies

of

Lakehead University, Thunder Bay

Committee in charge:

Dr. Vijay Mago (Principal Supervisor)

Dr. Thangarajah Akilan (External Examiner)
Dr. Quazi Abidur Rahman (Internal Examiner)

Winter 2021

The thesis of Dhivya Chandrasekaran, titled Exploiting semantic similarity models to automate transfer credit assessment in academic mobility, is approved:

Chair	______________________	Date	__________
	______________________	Date	__________
	______________________	Date	__________

Lakehead University, Thunder Bay

Exploiting semantic similarity models to automate transfer credit assessment in academic mobility

Abstract

EXPLOITING SEMANTIC SIMILARITY MODELS TO AUTOMATE TRANSFER CREDIT ASSESSMENT IN ACADEMIC MOBILITY

Student mobility or academic mobility involves students moving between institutions during their post-secondary education, and one of the challenging tasks in this process is to assess the transfer credits to be offered to the incoming student. In general, this process involves domain experts comparing the learning outcomes (LOs) of the courses, and based on their similarity deciding on offering transfer credits to the incoming students. This manual implementation of the task is not only labor-intensive but also influenced by undue bias and administrative complexity. This research work focuses on identifying an algorithm that exploits the advancements in the field of Natural Language Processing (NLP) to effectively automate this process. A survey tracing the evolution of semantic similarity helps understand the various methods available to calculate the semantic similarity between text data. The basic units of comparison namely, learning outcomes are made up of two components namely the descriptor part which provides the contents covered, and the action word which provides the competency achieved. Bloom's taxonomy provides six different levels of competency to which the action words fall into. Given the unique structure, domain specificity, and complexity of learning outcomes, a need for designing a tailor-made algorithm arises. The proposed algorithm uses a clustering-inspired methodology based on knowledge-based semantic similarity measures to assess the taxonomic similarity of learning outcomes and a transformer-based semantic similarity model to assess the semantic similarity of the learning outcomes. The cumulative similarity between the learning outcomes is further aggregated to form course to course similarity. Due to the lack of quality benchmark datasets, a new benchmark dataset is built by conducting a survey among domain experts with knowledge in both academia and computer science. The dataset contains 7 course-to-course similarity values annotated by 5 domain experts. Understanding the inherent need for flexibility in the decision-making process the aggregation part of the algorithm offers tunable parameters to accommodate different scenarios. Being one of the early research works in the field of automating articulation, this thesis establishes the imminent challenges that need to be addressed in the field namely, the significant decrease in performance by state-of-the-art semantic similarity models with an increase in complexity of sentences, lack of large datasets to train/fine-tune existing models, lack of quality in available learning outcomes, and reluctance to share learning outcomes publicly. While providing an efficient algorithm to assess the similarity between courses with existing resources, this research work steers future research attempts to apply NLP in the field of articulation in an ideal direction by highlighting the persisting research gaps.

Dedication

To my grandmother, Late Mrs. Susila Ramanunjam, for teaching me the value of education, inspiring me to dream big and to work relentlessly to achieve that dream.

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to praise and thank the Almighty God Jesus for His abundant Grace and Strength that enabled me to pursue my research successfully. I would like to extend my sincere gratitude to my supervisor Dr. Vijay Mago for his academic, financial, and moral support without which this research would not have been possible, for his motivation and confidence in me that helped me cross many hurdles with ease. My sincere thanks to my project manager Andrew Heppner for his mentorship and support. I would like to acknowledge the financial support from the Ontario Council on Articulation and Transfer. I would like to thank my teammates in DaTALab for their inspiration, support, and all the fun times. Last but not the least, I would like to thank my husband who stood by my dreams, my family, and my friends for their unconditional love and affection.

Chapter 1

Introduction

Natural language processing (NLP) has seen significant breakthroughs in recent years with the introduction of transformer-based language models which use attention mechanisms to build contextual embeddings. One of the most researched tasks in natural language processing is semantic textual similarity, which attempts to assess the similarity between two text snippets. This thesis work explores extensively the existing semantic similarity methods by conducting a survey of more than a hundred research articles published in reputed venues over the last three decades. The semantic similarity models are classified into knowledge-based, corpus-based, deep neural network based, and hybrid methods based on the underlying principle used to measure the similarity. While the proposed models achieve near-perfect results in existing datasets, this research work explores one of the major shortcomings of these models where the performance of the recent transformer models significantly decreases with the increase in complexity of the sentences. The complexity of the sentences in the existing popular benchmark datasets are analyzed and owing to their simplicity, a new benchmark dataset comprising of complex, domain-specific sentences is proposed and the performance of the popular BERT-variants is analyzed. Understanding the advantages of the semantic similarity models and their limitations, this research work explores the application of these models in a real world task - student mobility. The proposed model uses a transformer-based language model to assess the semantic similarity between learning outcomes and a clustering-inspired algorithm to assess their taxonomic similarity. Finally, the model uses a straightforward and transparent aggregation process to determine the course level similarity and hence assess the transfer credit for the incoming students. The final chapter concludes by highlighting the shortcomings of the semantic similarity models and the challenges in the domain of articulation.

1.1 Thesis organization

This thesis is further organized as follows. Chapter 2 discusses in detail the various semantic similarity methods classifying them as knowledge-based (Chapter 2.2), corpus-based (Chapter 2.3), deep neural network-based (Chapter 2.4), and hybrid (Chapter 2.5), also discussing their advantages and disadvantages. Chapter 3 proposes a new Domain Specific Complex Sentence (DSCS) dataset which highlights one of the drawback of the existing semantic similarity models. Chapter 4 presents an algorithm to measure the transfer credit for courses based on the semantic and taxonomic similarity of learning outcomes. Chapter 5 provides a comprehensive conclusion of the research work.

Chapter 2

Literature review - Semantic Similarity

This chapter contains excerpts from the article published in the following peer reviewed journal:

- Chandrasekaran D. & Mago, V. (2021).Text Evolution of Semantic Similarity—A Survey. ACM Computing Surveys (CSUR), 54(2), 1-37.

In order to effectively understand the existing semantic similarity methods and their classifications, an extensive survey was conducted and compiled to form the survey article mentioned above. In this chapter content from the article has been added, removed, or summarized, where appropriate.

2.1 Introduction

With the exponential increase in text data generated over time, Natural Language Processing (NLP) has gained significant attention from Artificial Intelligence (AI) experts. Measuring the semantic similarity between various text components like words, sentences, or documents plays a significant role in a wide range of NLP tasks like information retrieval [57], text summarization [94], text classification [58], essay evaluation [50], text simplification [142], machine translation [166], question answering [22, 79], among others. In the early days, two text snippets were considered similar if they contain the same words/characters. The techniques like Bag of Words (BoW), Term Frequency - Inverse Document Frequency (TF-IDF) were used to represent text, as real value vectors to aid calculation of semantic similarity. However, these techniques did not attribute to the fact that words have different meanings and different words can be used to represent a similar concept. For example, consider two sentences *"John and David studied Maths and Science."* and *"John studied Maths and David studied Science."* Though these two sentences have exactly the same words they do not convey the same meaning. Similarly, the sentences *"Mary is allergic to dairy products."* and *"Mary is lactose intolerant."* convey the same meaning; however, they do not have the same set of words. These methods captured the lexical feature of the text and were simple to implement, however, they ignored the semantic and syntactic properties of text. To address these drawbacks of the lexical measures various semantic similarity techniques were proposed over the past three decades.

Semantic Textual Similarity (STS) is defined as the measure of semantic equivalence between two blocks of text. Semantic similarity methods usually give a ranking or percentage of similarity between texts, rather than a binary decision as similar or not similar. Semantic similarity is often used synonymously with semantic relatedness. However, semantic relatedness not only accounts for the semantic similarity between texts but also considers a broader perspective analyzing the shared semantic properties of two words. For example, the words *'coffee'* and *'mug'* may be related to one another closely, but they are not considered semantically similar whereas the words *'coffee'* and *'tea'* are semantically similar. Thus, semantic similarity may be considered, as one of the aspects of semantic relatedness. The semantic relationship including similarity is measured in terms of semantic distance, which is inversely proportional to the relationship [44].

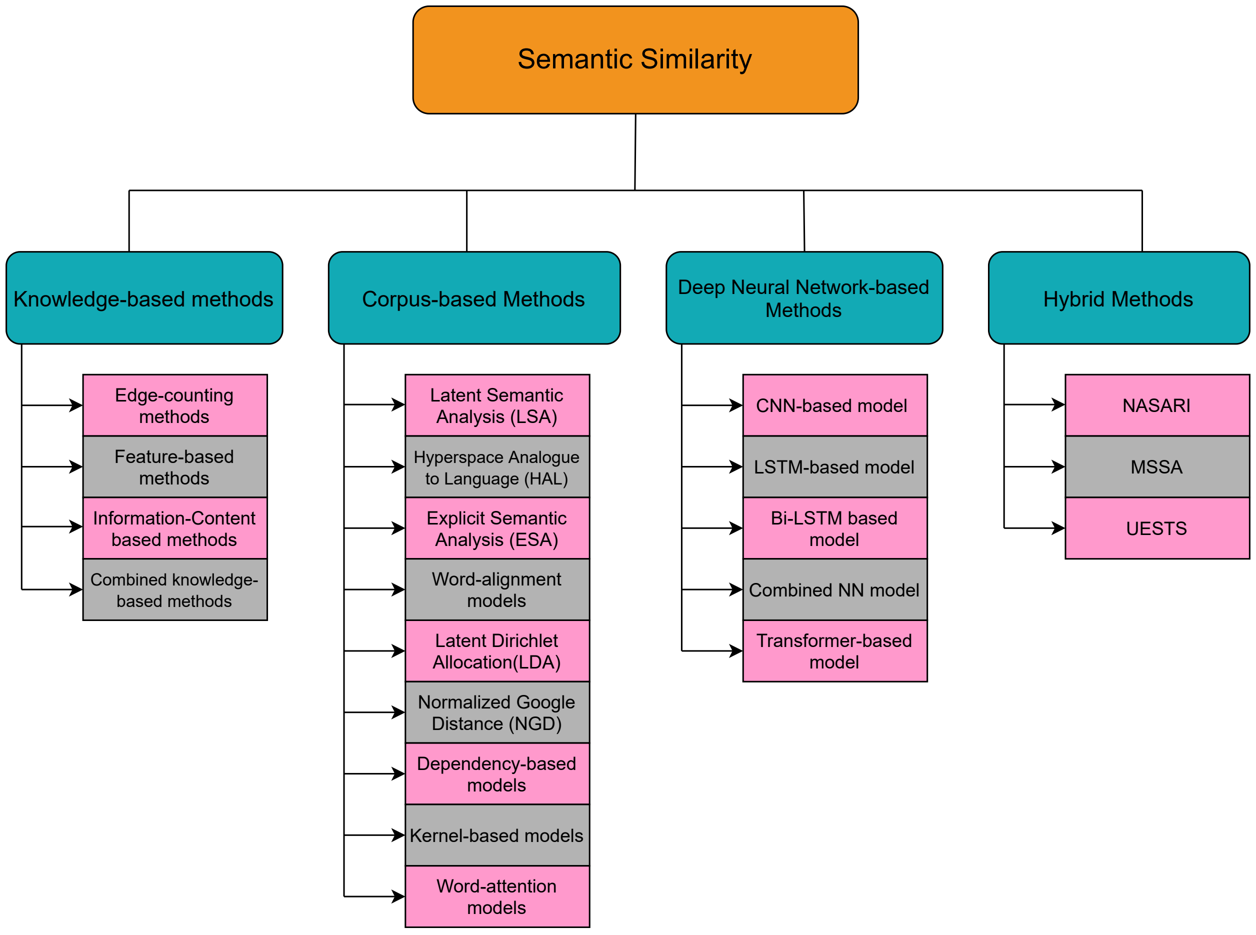


Figure 2.1: Classification of Semantic similarity methods.

Chapter Organization

Most of the survey articles published recently related to semantic similarity, provide in-depth knowledge of one particular semantic similarity technique or a single application of semantic similarity. Lastra-Díaz et al. survey various knowledge-based methods [63] and IC-based methods [64], Camacho-Colladas et al. [23] discuss various vector representation methods of words, Taieb et al. [44], on the other hand, describe various semantic relatedness methods and Berna Altınel et al. [9] summarise various semantic similarity methods used for text classification. However, this chapter provides a comprehensive account of the various semantic similarity techniques including the most recent advancements using deep neural network-based methods.

This chapter traces the evolution of Semantic Similarity Techniques over the past decades, distinguishing them based on the underlying methods used in them. Figure 2.1 shows the structure of the chapter. Sections 2.2 to 2.5 provide a detailed description of semantic similarity methods broadly classified as 1) Knowledge-based methods, 2) Corpus-based methods, 3) Deep neural network-based methods, and 4) Hybrid methods. The various datasets mentioned in this chapter are discussed in detail in Appendix B. This chapter provides a deep and wide knowledge of existing techniques for new researchers who venture to explore one of the most challenging NLP tasks, Semantic Textual Similarity.

2.2 Knowledge-based methods

Knowledge-based semantic similarity methods calculate semantic similarity between two terms based on the information derived from one or more underlying knowledge sources like ontologies/lexical databases, thesauri, dictionaries, etc. The underlying knowledge-base offers these methods a structured representation of terms or concepts connected by semantic relations, further offering an ambiguity free semantic measure, as the actual meaning of the terms, is taken into consideration [132]. In this section, we discuss four lexical databases widely employed in knowledge-based semantic similarity methods and further discuss in brief, different methodologies adopted by some of the knowledge-based semantic similarity methods.

Lexical Databases

- WordNet [91] is a widely used lexical database for knowledge-based semantic similarity methods that accounts for more than 100,000 English concepts [132]. WordNet can be visualized as a graph, where the nodes represent the meaning of the words (concepts), and the edges define the relationship between the words [164]. WordNet's structure is primarily based on synonyms, where each word has different *synsets* attributed to their different meanings. The similarity between two words depends on the path distance between them [108].

- Wiktionary[1] is an open-source lexical database that encompasses approximately 6.2 million words from 4,000 different languages. Each entry has an article page associated with it, and it accounts for a different sense of each entry. Wiktionary does not have a well-established taxonomic lexical relationship within the entries, unlike WordNet, which makes it difficult to be used in semantic similarity algorithms [114].

- With the advent of Wikipedia[2], most techniques for semantic similarity exploit the abundant text data freely available to train the models [88]. Wikipedia has the text

[1]https://en.wiktionary.org
[2]http://www.wikipedia.org

data organized as Articles. Each article has a title (concept), neighbors, description, and categories. It is used as both structured taxonomic data and/or as a corpus for training corpus-based methods [115]. The complex category structure of Wikipedia is used as a graph to determine the Information Content of concepts, which in turn aids in calculating the semantic similarity [53].

- BabelNet [102] is a lexical resource that combines WordNet with data available on Wikipedia for each *synset*. It is the largest multilingual semantic ontology available with nearly over 13 million *synsets* and 380 million semantic relations in 271 languages. It includes over four million *synsets* with at least one associated Wikipedia page for the English language [25].

Types of Knowledge-based semantic similarity methods

Based on the underlying principle of how the semantic similarity between words is assessed, knowledge-based semantic similarity methods can be further categorized as edge-counting methods, feature-based methods, and information content-based methods.

Edge-counting methods:

The most straight forward edge counting method is to consider the underlying ontology as a graph connecting words taxonomically and count the edges between two terms to measure the similarity between them. The greater the distance between the terms the less similar they are. This measure called *path* was proposed by Rada et al. [118] where the similarity is inversely proportional to the shortest path length between two terms. In this edge-counting method, the fact that the words deeper down the hierarchy have a more specific meaning, and that, they may be more similar to each other even though they have the same distance

as two words that represent a more generic concept was not taken into consideration. Wu and Palmer [161] proposed *wup* measure, where the depth of the words in the ontology was considered an important attribute. The *wup* measure counts the number of edges between each term and their Least Common Subsumer (LCS). LCS is the common ancestor shared by both terms in the given ontology. Consider, two terms denoted as t_1, t_2, their LCS denoted as t_{lcs}, and the shortest path length between them denoted as $min\_len(t_1, t_2)$,
path is measured as,

$$sim_{path}(t_1, t_2) = \frac{1}{1 + min\_len(t_1, t_2)} \tag{2.1}$$

and *wup* is measured as,

$$sim_{wup}(t_1, t_2) = \frac{2depth(t_{lcs})}{depth(t_1) + depth(t_2)} \tag{2.2}$$

Li et al. [74] proposed a measure that takes into account both the minimum path distance and depth. *li* is measured as,

$$sim_{li} = e^{-\alpha min\_len(t_1,t_2)} . \frac{e^{\beta depth(t_{lcs})} - e^{-\beta depth(t_{lcs})}}{e^{\beta depth(t_{lcs})} + e^{-\beta depth(t_{lcs})}} \tag{2.3}$$

However, the edge-counting methods ignore the fact that the edges in the ontologies need not be of equal length. To overcome this shortcoming of simple edge-counting methods feature-based semantic similarity methods were proposed.

Feature-based methods:

The feature-based methods calculate similarity as a function of properties of the words, like gloss, neighboring concepts, etc. [132]. Gloss is defined as the meaning of a word in a dictionary; a collection of glosses is called a glossary. There are various semantic similarity methods proposed based on the gloss of words. Gloss-based semantic similarity measures exploit the knowledge that words with similar meanings have more common words in their gloss. The semantic similarity is measured as the extent of overlap between the gloss of the words in consideration. The Lesk measure [13], assigns a value of relatedness between two words based on the overlap of words in their gloss and the glosses of the concepts they are related to in an ontology like WordNet [63]. Jiang et al. [52] proposed a feature-based method where semantic similarity is measured using the glosses of concepts present in Wikipedia. Most feature-based methods take into account common and non-common features between two words/terms. The common features contribute to the increase of the similarity value and the non-common features decrease the similarity value. The major limitation of feature-based methods is its dependency on ontologies with semantic features, and most ontologies rarely incorporate any semantic features other than taxonomic relationships [132].

Information Content-based methods:

Information content (IC) of a concept is defined as the information derived from the concept when it appears in context [130]. A high IC value indicates that the word is more specific and clearly describes a concept with less ambiguity, while lower IC values indicate that the words are more abstract in meaning [164]. The specificity of the word is determined using Inverse Document Frequency (IDF), which relies on the principle that the more specific a word is, the less it occurs in a document. Information content-based methods measure the similarity between terms using the IC value associated with them. Resnik and Philip [124] proposed a semantic similarity measure called *res* which measures the similarity based on the idea that if two concepts share a common subsumer they share more information since the *IC* value of the LCS is higher. Considering *IC* represents the Information Content of the given term, *res* is measured as,

$$sim_{res}(t_1, t_2) = IC_{t_{lcs}} \tag{2.4}$$

D. Lin [76] proposed an extension of the res measure taking into consideration the IC value of both the terms that attribute to the individual information or description of the terms and the IC value of their LCS that provides the shared commonality between the terms. lin is measured as,

$$sim_{lin}(t_1, t_2) = \frac{2IC_{t_{lcs}}}{IC_{t_1} + IC_{t_2}} \tag{2.5}$$

Jiang and Conrath [51] calculate a distance measure based on the difference between the sum of the individual IC values of the terms and the IC value of their LCS using the below equation,

$$dis_{jcn}(t_1, t_2) = IC_{t_1} + IC_{t_2} - 2ICt_{lcs} \tag{2.6}$$

The distance measure replaces the shortest path length in equation (1), and the similarity is inversely proportional to the above distance. Hence jcn is measured as,

$$sim_{jcn}(t_1, t_2) = \frac{1}{1 + dis_{jcn}(t_1, t_2)} \tag{2.7}$$

IC can be measured using an underlying corpora or from the intrinsic structure of the ontology itself [131] based on the assumption that the ontologies are structured in a meaningful way. Some of the terms may not be included in one ontology, which provides a scope to use multiple ontologies to calculate their relationship [125]. Based on whether the given terms are both present in a single ontology or not, IC-based methods can be classified as mono-ontological methods or multi-ontological methods. When multiple ontologies are involved the IC of the Least Common Subsumer from both the ontologies are accessed to estimate the semantic similarity values. Jiang et al. [53] proposed IC-based semantic similarity measures based on Wikipedia pages, concepts and neighbors. Wikipedia was both used as a structured taxonomy as well as a corpus to provide IC values.

Combined knowledge-based methods:

Various similarity measures were proposed combining the various knowledge-based methods. Goa et al. [39] proposed a semantic similarity method based on WordNet ontology where three different strategies are used to add weights to the edges and the shortest weighted path is used to measure the semantic similarity. According to the first strategy, the depths of all the terms in WordNet along the path between the two terms in consideration is added as a weight to the shortest path. In the second strategy, only the depth of the LCS of the terms was added as the weight, and in strategy three, the IC value of the terms is added as weight. The shortest weighted path length is now calculated and then non-linearly transformed to produce semantic similarity measures. In comparison, it is shown that strategy three achieved a better correlation to the gold standards in comparison with traditional methods and the two other strategies proposed. Zhu and Iglesias [164] proposed another weighted path measure called *wpath* that adds the IC value of the Least Common Subsumer as a weight to the shortest path length. *wpath* is calculated as

$$sim_{wpath}(t_1, t_2) = \frac{1}{1 + min\_len(t_1, t_2) * k^{IC_{t_{lcs}}}} \tag{2.8}$$

This method was proposed to be used in various knowledge graphs (KG) like WordNet [91], DBPedia [19], YAGO [49], etc. and the parameter k is a hyperparameter which has to be tuned for different KGs and different domains as different KGs have a different distribution of terms in each domain. Both corpus-based IC and intrinsic IC values were experimented and corpus IC-based *wpath* measure achieved greater correlation in most of the gold standard datasets.

Knowledge-based semantic similarity methods are computationally simple, and the underlying knowledge-base acts as a strong backbone for the models, and the most common problem of ambiguity like synonyms, idioms, and phrases are handled efficiently. Knowledge-based methods can easily be extended to calculate sentence to sentence similarity measure by defining rules for aggregation [70]. Lastra-Díaz et al. [65] developed a software Half-Edge Semantic Measures Library (HESML) to implement various ontology-based semantic similarity measures proposed and have shown an increase in performance time and scalability of the models.

However, knowledge-based systems are highly dependent on the underlying source resulting in the need to update them frequently which requires time and high computational resources. Although strong ontologies like WordNet, exist for the English language, similar resources are not available for other languages that results in the need for the building of strong and structured knowledge bases to implement knowledge-based methods in different languages and across different domains. Various research works were conducted on extending semantic similarity measures in the biomedical domain [109, 145]. McInnes et al. [85] built a domain-specific model called UMLS to measure the similarity between words in the biomedical domain. With nearly 6,500 world languages and numerous domains, this becomes a serious drawback for knowledge-based systems.

2.3 Corpus-based methods

Corpus-based semantic similarity methods measure semantic similarity between terms using the information retrieved from large corpora. The underlying principle called 'distributional hypothesis' [42] exploits the idea that "similar words occur together, frequently"; however, the actual meaning of the words is not taken into consideration. While various techniques were used to construct the vector representation of the text data, several semantic distance measures based on the distributional hypothesis were proposed to estimate the similarity between the vectors. A comprehensive survey of various distributional semantic measures was carried out by Mohammad and Hurst [95], and the different measure and their respective formula are provided in Table C.1 in Appendix C. However, among all these measures, the cosine similarity gained significance and has been widely used among NLP researchers to date [95]. In this section, we discuss in detail some of the widely used word-embeddings built using distributional hypothesis and some of the significant corpus-based semantic similarity methods.

Word Embeddings

Word embeddings provide vector representations of words wherein these vectors retain the underlying linguistic relationship between the words [135]. These vectors are computed using different approaches like neural networks [89], word co-occurrence matrix [110], or representations in terms of the context in which the word appears [71]. Some of the most widely used pre-trained word embeddings include:

- ***word2vec*** [89]: Developed from Google News dataset, containing approximately 3 million vector representations of words and phrases, *word2vec* is a neural network model used to produce distributed vector representation of words based on an underlying corpus. There are two different models of *word2vec* proposed: the Continuous Bag of Words (CBOW) and the Skip-gram model. The architecture of the network is rather simple and contains an input layer, one hidden layer, and an output layer. The network is fed with a large text corpus as the input, and the output of the model is the vector representations of words. The CBOW model predicts the current word using the neighboring context words, while the Skip-gram model predicts the neighboring context words given a target word. *word2vec* models are efficient in representing the words as vectors that retain the contextual similarity between words. The word vector calculations yielded good results in predicting the semantic similarity [90]. Many researchers extended the *word2vec* model to propose context vectors [87], dictionary vectors [156], sentence vectors [106] and paragraph vectors [66].

- ***GloVe*** [110]: *GloVe* developed by Stanford University relies on a global word co-occurrence matrix formed based on the underlying corpus. It estimates similarity based on the principle that words similar to each other occur together. The co-occurrence matrix is populated with occurrence values by doing a single pass over the underlying large corpora. *GloVe* model was trained using five different corpora mostly Wikipedia dumps. While forming vectors, words are chosen within a specified context window owing to the fact that words far away have less relevance to the context word in consideration. The *GloVe* loss function minimizes the least-square distance between the context window co-occurrence values and the global co-occurrence values [63]. *GloVe* vectors were extended to form contextualized word vectors to differentiate words based on context [84].

- ***fastText*** [21]: Facebook AI researchers developed a word embedding model that builds word vectors based on Skip-gram models where each word is represented as a collection of character n-grams. *fastText* learns word embeddings as the average of its character embeddings thus accounting for the morphological structure of the word which proves efficient in various languages like Finnish and Turkish. Even out-of-the-vocabulary words are assigned word vectors based on their characters or subunits.

- ***Bidirectional Encoder Representations from Transformers(BERT)*** [35]: Devlin et al. [35] proposed a pretrained transformer-based word embeddings which can be fine-tuned by adding a final output layer to accommodate the embeddings to different NLP tasks. BERT uses the transformer architecture proposed by Vaswani et al.

[157], which produces attention-based word vectors using a bi-directional transformer encoder. The BERT framework involves two important processes namely 'pre-training' and 'fine-tuning'. The model is pretrained using a corpus of nearly 3,300M words from both the Book corpus and English Wikipedia. Since the model is bidirectional in order to avoid the possibility of the model knowing the token itself when training from both directions the pretraining process is carried out in two different ways. In the first task, random words in the corpus are masked and the model is trained to predict these words. In the second task, the model is presented with sentence pairs from the corpus, in which 50 percent of the sentences are actually consecutive while the remaining are random pairs. The model is trained to predict if the given sentence pair are consecutive or not. In the 'fine-tuning' process, the model is trained for the specific down-stream NLP task at hand. The model is structured to take as input both single sentences and multiple sentences to accommodate a variety of NLP tasks. To train the model to perform a question answering task, the model is provided with various question-answer pairs and all the parameters are fine-tuned in accordance with the task. BERT embeddings provided state-of-the-art results in the STS-B data set with a Spearman's correlation of 86.5% outperforming other BiLSTM models including ELMo [111].

Word embeddings are used to measure semantic similarity between texts of different languages by mapping the word embedding of one language over the vector space of another. On training with a limited yet sufficient number of translation pairs, the translation matrix can be computed to enable the overlap of embeddings across languages [41]. One of the major challenges faced when deploying word-embeddings to measure similarity is Meaning Conflation Deficiency. It denotes that word embeddings do not attribute to the different meanings of a word that pollutes the semantic space with noise by bringing irrelevant words closer to each other. For example, the words 'finance' and 'river' may appear in the same semantic space since the word 'bank' has two different meanings [23]. It is critical to understand that word-embeddings exploit the distributional hypothesis for the construction of vectors and rely on large corpora, hence, they are classified under corpus-based semantic similarity methods. However, deep-neural network based-methods and most hybrid semantic similarity methods use word-embeddings to convert the text data to high dimensional vectors, and the efficiency of these embeddings plays a significant role in the performance of the semantic similarity methods [93, 72].

Types of corpus-based semantic similarity methods

Based on the underlying methods using which the word-vectors are constructed there are a wide variety of corpus-based methods some of which are discussed in this section.

Latent Semantic Analysis (LSA) [61]:

LSA is one of the most popular and widely used corpus-based techniques used for measuring semantic similarity. A word co-occurrence matrix is formed where the rows represent the words and columns represent the paragraphs, and the cells are populated with word counts. This matrix is formed with a large underlying corpus, and dimensionality reduction is achieved by a mathematical technique called Singular Value Decomposition (SVD). SVD represents a given matrix as a product of three matrices, where two matrices represent the rows and columns as vectors derived from their eigenvalues and the third matrix is a diagonal matrix that has values that would reproduce the original matrix when multiplied with the other two matrices [62]. SVD reduces the number of columns while retaining the number of rows thereby preserving the similarity structure among the words. Then each word is represented as a vector using the values in its corresponding rows and semantic similarity is calculated as the cosine value between these vectors. LSA models are generalized by replacing words with texts and columns with different samples and are used to calculate the similarity between sentences, paragraphs, and documents.

Hyperspace Analogue to Language(HAL) [81]:

HAL builds a word co-occurrence matrix that has both rows and columns representing the words in the vocabulary and the matrix elements are populated with association strength values. The association strength values are calculated by sliding a "window" the size of which can be varied, over the underlying corpus. The strength of association between the words in the window decreases with the increase in their distance from the focused word. For example, in the sentence "This is a survey of various semantic similarity measures", the words *'survey'* and *'variety'* have greater association value than the words *'survey'* and *'measures.'* Word vectors are formed by taking into consideration both the row and column of the given word. Dimensionality reduction is achieved by removing any columns with low entropy values. The semantic similarity is then calculated by measuring the Euclidean or Manhattan distance between the word vectors.

Explicit Semantic Analysis (ESA) [37]:

ESA measures semantic similarity based on Wiki-pedia concepts. The use of Wikipedia ensures that the proposed method can be used over various domains and languages. Since Wikipedia is constantly updated, the method is adaptable to the changes over time. First, each concept in Wikipedia is represented as an attribute vector of the words that occur in it, then an inverted index is formed, where each word is linked to all the concepts it is associated with. The association strength is weighted using the TF-IDF technique, and the concepts weakly associated with the words are removed. Thus the input text is represented by weighted vectors of concepts called the "interpretation vectors." Semantic similarity is measured by calculating the cosine similarity between these word vectors.

Word-Alignment models [149]:

Word-Alignment models calculate the semantic similarity of sentences based on their alignment over a large corpus [148, 56, 27]. The second, third, and fifth positions in SemEval tasks 2015 were secured by methods based on word alignment. The unsupervised method which was in the fifth place implemented the word alignment technique based on Paraphrase Database (PPDB) [38]. The system calculates the semantic similarity between two sentences as a proportion of the aligned context words in the sentences over the total words in both the sentences. The supervised methods which were at the second and third place used *word2vec* to obtain the alignment of the words. In the first method, a sentence vector is formed by computing the "component-wise average" of the words in the sentence, and the cosine similarity between these sentence vectors is used as a measure of semantic similarity. The second supervised method takes into account only those words that have a contextual semantic similarity [149].

Latent Dirichlet Allocation (LDA) [144]:

LDA is used to represent a topic or the general idea behind a document as a vector rather than every word in the document. This technique is widely used for topic modeling tasks and it has the advantage of reduced dimensionality considering that the topics are significantly less than the actual words in a document [144]. One of the novel approaches to determine document-to-document similarity is the use of vector representation of documents and calculate the cosine similarity between the vectors to ascertain the semantic similarity between documents [18].

Normalised Google Distance [30]:

NGD measures the similarity between two terms based on the results obtained when the terms are queried using the Google search engine. It is based on the assumption that two words occur together more frequently in web-pages if they are more related. Give two terms t_1 and t_2 the following formula is used to calculate the NGD between the two terms.

$$NGD(x,y) = \frac{max\ \{log\ f(t_1), log\ f(t_2)\} - log\ f(t_1,t_2)}{log\ G - min\ \{log\ f(t_1), log\ f(t_2)\}} \tag{2.9}$$

where the functions $f(x)$ and $f(y)$ return the number of hits in Google search of the given terms, $f(x,y)$ returns the number of hits in Google search when the terms are searched together and G represent the total number of pages in the overall google search. NGD is widely used to measure semantic relatedness rather than semantic similarity because related terms occur together more frequently in web pages though they may have opposite meaning.

Dependency-based models [2]:

Dependency-based approaches ascertain the meaning of a given word or phrase using the neighbors of the word within a given window. The dependency-based models initially parse the corpus based on its distribution using Inductive Dependency Parsing [104]. For every given word a "syntactic context template" is built considering both the nodes preceding and succeeding the word in the built parse tree. For example, the phrase *"thinks ¡term¿ delicious"* could have a context template as *"pizza, burger, food"*. Vector representation of a word is formed by adding each window across the location that has the word in consideration, as it's root word, along with the frequency of the window of words appearing in the entire corpus. Once this vector is formed semantic similarity is calculated using cosine similarity between these vectors. Levy et al. [71] proposed DEPS embedding as a word-embedding model based on dependency-based bag of words. This model was tested with the WS353 dataset where the task was to rank the similar words above the related words. On plotting a recall precision curve the DEPS curve showed greater affinity towards similarity rankings over BoW methods taken in comparison.

Kernel-based models [141]:

Kernel-based methods were used to find patterns in text data thus enabling detecting similarity between text snippets. Two major types of kernels were used in text data namely the string or sequence kernel [26] and the tree kernel [98]. Moschitti et al. [98] proposed tree kernels in 2007, that contains three different sub-structures in the tree kernel space namely a subtree - a tree whose root is not a leaf node along with its children nodes, a subset tree - a tree whose root is not a leaf node but not incorporating all its children nodes and does not break the grammatical rules, a partial tree - a tree structure closely similar to subset tree but it doesn't always follow the grammatical rules. Tree kernels are widely used in identifying a structure in input sentences based on constituency or dependency, taking into consideration the grammatical rules of the language. Kernels are used by machine learning algorithms like Support Vector Machines(SVMs) to adapt to text data in various tasks like Semantic Role Labelling, Paraphrase Identification [34], Answer Extraction [99], Question-Answer classification [101], Relational text categorization [97], Answer Re-ranking in QA tasks [137] and Relational text entailment [100]. Severyn et al. [138] proposed a kernel-based semantic similarity method that represents the text directly as "structural objects" using Syntactic tree kernel [33] and Partial tree kernels [96]. The kernel function then combines the tree structures with semantic feature vectors from two of the best performing models in STS 2012 namely UKP [14] and Takelab [134] and some additional features including cosine similarity scores based on named entities, part of speech tags, and so on. The authors compare the performance of the model constructed using four different tree structures namely shallow tree, constituency tree, dependency tree, phrase-dependency tree, and the above-mentioned feature vectors. They establish that the tree kernel models perform better than all feature vectors combined. The model uses Support Vector Regression to obtain the final similarity score and it can be useful in various downstream NLP applications like question-answering, text-entailment extraction, etc. Amir et al. [10] proposed another semantic similarity algorithm using kernel functions. They used constituency-based tree kernels where the sentence is broken down into subject, verb, and object based on the assumption most semantic properties of the sentence are attributed to these components. The input sentences are parsed using the Stanford Parser to extract various combinations of subject, verb, and object. The similarity between the various components of the given sentences is calculated using a knowledge base, and different averaging techniques are used to average the similarity values to estimate the overall similarity, and the best among them is chosen based on the root mean squared error value for a particular dataset. In recent research, deep learning methods have been used to replace the traditional machine learning models and efficiently use the structural integrity of kernels in the embedded feature extraction stage [34, 32]. The model which achieved the best results in SemEval-2017 Task 1, proposed by Tian et al. [154] uses kernels to extract features from text data to calculate similarity. The model proposed an ensemble model that used both traditional NLP methods and deep learning methods. Two different features are namely the sentence pair matching features and single sentence features were used to predict the similarity values using regressors which added nonlinearity to the prediction. In single sentence feature extraction, dependency-based tree kernels are used to extract the dependency features in one given sentence, and in sentence pair matching features, constituency-based parse tree kernels are used to find the common sub-constructs among the three different characterizations of tree kernel spaces. The final similarity score is accessed by averaging the traditional NLP similarity value and the deep learning-based similarity value. The model achieved a Pearson's correlation of 73.16% in the STS dataset.

Word-attention models [67]:

In most of the corpus-based methods all text components are considered to have equal significance; however, human interpretation of measuring similarity usually depends on keywords in a given context. Word attention models capture the importance of the words from underlying corpora [80] before calculating the semantic similarity. Different techniques like word frequency, alignment, word association are used to capture the attention-weights of the text in consideration. Attention Constituency Vector Tree (ACV-Tree) proposed by Le et al. [67] is similar to a parse tree where one word of a sentence is made the root and the remainder of the sentence is broken as a Noun Phrase (NP) and a Verb Phrase (VP). The nodes in the tree store three different attributes of the word into consideration: the word vector determined by an underlying corpus, the attention-weight, and the "modification-relations" of the word. The modification relations can be defined as the adjectives or adverbs that modify the meaning of another word. All three components are linked to form the representation of the word. A tree kernel function is used to determine the similarity between two words based on the equation below

$$TreeKernel(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2) \tag{2.10}$$

$$\Delta(n_1, n_2) = \begin{cases} 0, \text{ if } (n_1 \text{ and / or } n_2 \text{ are non-leaf-nodes) and } n_1 \neq n_2 \\ Aw \times SIM(vec_1, vec_2), \text{ if } n_1, n_2 \text{are leaf nodes} \\ \mu(\lambda^2 + \sum_{p=1}^{l_m} \delta_p(c_{n_1}, c_{n_2})), \text{ otherwise} \end{cases} \tag{2.11}$$

where n_1, n_2 represent the represents the nodes, $SIM(vec_1, vec_2)$ measures the cosine similarity between the vectors, $\delta_p(.)$ calculates the number of common subsequences of length p, λ, μ denote the decay factors for length of the child sequences and the height of the tree respectively, c_{n_1}, c_{n_2} refer to the children nodes and $l_m = min(length(c_{n_1}), length(c_{n_2}))$. The algorithm is tested using the STS benchmark datasets and has shown better performance in 12 out of 19 chosen STS Datasets [67, 116].

Unlike knowledge-based systems, corpus-based systems are language and domain independent [9]. Since they are dependent on statistical measures the methods can be easily adapted across various languages using an effective corpus. With the growth of the internet, building corpora of most languages or domains has become rather easy. Simple web crawling techniques can be used to build large corpora [16]. However, the corpus-based methods do not take into consideration the actual meaning of the words. The other challenge faced by corpus-based methods is the need to process the large corpora built, which is a rather time-consuming and resource-dependent task. Since the performance of the algorithms largely depends on the underlying corpus, building an efficient corpus is paramount. Though efforts are made by researchers to build a clean and efficient corpus like the C4 corpus built by web crawling and five steps to clean the corpus [121], an "ideal corpus" is still not defined by researchers.

2.4 Deep neural network-based methods

Semantic similarity methods have exploited the recent developments in neural networks to enhance performance. The most widely used techniques include Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), Bidirectional Long Short Term Memory (Bi-LSTM), and Recursive Tree LSTM. Deep neural network models are built based on two fundamental operations: convolution and pooling. The convolution operation in text data may be defined as the sum of the element-wise product of a sentence vector and a weight matrix. Convolution operations are used for feature extraction. Pooling operations are used to eliminate features that have a negative impact, and only consider those feature values that have a considerable impact on the task at hand. There are different types of pooling operations and the most widely used is Max pooling, where only the maximum value in the given filter space is selected. This section describes some of the methods that deploy deep neural networks to estimate semantic similarity between text snippets. Although the methods described below exploit word embeddings built using large corpora, deep-neural networks are used to estimate the similarity between the word-embeddings, hence they are classified separately from corpus-based methods.

Types of deep neural network-based semantic similarity methods:

- Wang et al. [159] proposed a model to estimate semantic similarity between two sentences based on lexical decomposition and composition. The model uses *word2vec* pretrained embeddings to form a vector representation of the sentences s_1 and s_2. A similarity matrix M of dimension i x j is built where i and j are the number of words in sentence 1 (S_1) and sentence 2 (S_2) respectively. The cells of the matrix are populated with the cosine similarity between the words in the indices of the matrix. Three different functions are used to construct semantic matching vectors $\vec{s_1}$ and $\vec{s_2}$, the global, local, and max function. The global function constructs the semantic matching vector of S_1 by taking the weighted sum of the vectors, of all the words in S_2, the local function, takes into consideration only word vectors within a given window size, and the max function takes only the vectors of the words, that have the maximum similarity. The second phase of the algorithm uses three different decomposition functions - rigid, linear, and orthogonal - to estimate the similarity component and the dissimilarity component between the sentence vectors and the semantic matching vectors. Both the similarity component and the dissimilarity component vectors are passed through a two-channel convolution layer followed by a single max-pooling layer. The similarity is then calculated using a sigmoid layer that estimates the similarity value within the range of 0 and 1. The model was tested using the QASent dataset [158] and the WikiQA dataset [86]. The two measures used to estimate the performance are mean average precision (MAP) and mean reciprocal rank (MRR). The model achieves the best MAP in the QASent dataset and the best MAP and MRR in the WikiQA dataset. Yang Shao [139] proposed a semantic similarity algorithm that exploits, the recent development in neural networks using $GloVe$ word embeddings. Given two sentences, the model predicts a probability distribution over set semantic similarity values. The pre-processing steps involve the removal of punctuation, tokenization, and using $GloVe$ vectors to replace words with word embeddings. The length of the input is set to 30 words, which is achieved by removal or padding as deemed necessary. Some special hand-crafted features like flag values indicating if the words or numbers occurred in both the sentences and POS tagging one hot encoded values, were added to the $GloVe$ vectors. The vectors are then fed to a CNN with 300 filters and one max-pooling layer which is used to form the sentence vectors. ReLU activation function is used in the convolution layer. The semantic difference between the vectors is calculated by the element-wise absolute difference and the element-wise multiplication of the two, sentence-vectors generated. The vectors are further passed through two fully-connected layers, which predicts the probability distribution of the semantic similarity values. The model performance was evaluated using the SemEval datasets where the model was ranked 3rd in SemEval 2017 dataset track.

- The LSTM networks are a special kind of Recurrent Neural Networks (RNN). While processing text data, it is essential for the networks to remember previous words, to capture the context, and RNNs have the capacity to do so. However, not all the previous content has significance over the next word/phrase, hence RNNs suffer the drawback of long term dependency. LSTMs are designed to overcome this problem. LSTMs have gates which enable the network to choose the content it has to remember. For example, consider the text snippet, *"Mary is from Finland. She is fluent in Finnish. She loves to travel."* While we reach the second sentence of the text snippet, it is essential to remember the words *"Mary"* and *"Finland."* However, on reaching the third sentence the network may forget the word *"Finland."* The architecture of LSTMs allows this. Many researchers use the LSTM architecture to measure semantic similarity between blocks of text. Tien et al. [155] uses a network combined with LSTM and CNN to form a sentence embedding from pretrained word embeddings followed by an LSTM architecture to predict their similarity. Tai et al. [152] proposed an LSTM architecture to estimate the semantic similarity between two given sentences. Initially, the sentences are converted to sentence representations using Tree-LSTM over the parse tree of the sentences. These sentence representations are then, fed to a neural network that calculates the absolute distance between the vectors and the angle between the vectors. The experiment was conducted using the SICK dataset, and the similarity measure varies with the range 1 to 5. The hidden layer consisted of 50 neurons and the final softmax layer classifies the sentences over the given range. The Tree-LSTM model achieved better Pearson's and Spearman's correlation in the gold standard datasets, than the other neural network models in comparison.

- He and Lin [46] proposed a hybrid architecture using Bi-LSTM and CNN to estimate the semantic similarity of the model. Bi-LSTMs have two LSTMs that run parallel, one from the beginning of the sentence and one from the end, thus capturing the entire context. In their model, He and Lin use Bi-LSTM for context modelling. A pairwise word interaction model is built that calculates a comparison unit between the vectors derived from the hidden states of the two LSTMs using the below formula

$$CoU(\vec{h_1}, \vec{h_2}) = \{cos(\vec{h_1}, \vec{h_2}), euc(\vec{h_1}, \vec{h_2}), manh((\vec{h_1}, \vec{h_2})\} \tag{2.12}$$

where $\vec{h_1}$ and $\vec{h_2}$ represent the vectors from the hidden state of the LSTMs and the functions $cos()$, $euc()$, $manh()$ calculate the Cosine distance, Euclidean distance, and Manhattan distance, respectively. This model is similar to other recent neural network-based word attention models [12, 8]. However, attention weights are not added, rather the distances are added as weights. The word interaction model is followed by a similarity focus layer where weights are added to the word interactions (calculated in the previous layers) based on their importance in determining the similarity. These re-weighted vectors are fed to the final convolution network. The network is composed of alternating spatial convolution layers and spatial max pooling layers, ReLU activation function is used and at the network ends with two fully connected layers followed by a LogSoftmax layer to obtain a non-linear solution. This model outperforms the previously mentioned Tree-LSTM model on the SICK dataset.

- Lopez-Gazpio et al. [80] proposed an extension to the existing Decomposable Attention Model (DAM) proposed by Parikh et al. [107] which was originally used for Natural Language Inference(NLI). NLI is used to categorize a given text block to a particular relation like entailment, neutral, or contradiction. The DAM model used feed-forward neural networks in three consecutive layers the attention layer, comparison layer, and aggregation layer. Given two sentences the attention layer produces two attention vectors for each sentence by finding the overlap between them. The comparison layer concatenates the attention vectors with the sentence vectors to form a single representative vector for each sentence. The final aggregation layer flattens the vectors and calculates the probability distribution over the given values. Lopez-Gazpio et al. [80] used word n-grams to capture attention in the first layer instead of individual words. $n-grams$ maybe defined as a sequence of n words that are contiguous with the given word, n-grams are used to capture the context in various NLP tasks. In order to accommodate n-grams, a Recurrent Neural Network (RNN) is added to the attention layer. Variations were proposed by replacing RNN with Long-Term Short memory (LSTM) and Convolutional Neural Network (CNN). The model was used for semantic similarity calculations by replacing the final classes of entailment relationships with semantic similarity ranges from 0 to 5. The models achieved better performance in capturing the semantic similarity in the SICK dataset and the STS benchmark dataset when compared to DAM and other models like Sent2vec [106] and BiLSTM among others.

- **Transformer-based models:** Vaswani et al. [157] proposed a transformer model that relies on attention mechanisms to capture the semantic properties of words in the embeddings. The transformer has two parts ‘encoder’ and ‘decoder’. The encoder consists of layers of multi-head attention mechanisms followed by a fully connected feed-forward neural network. The decoder is similar to the encoder with one additional layer of multi-head attention which captures the attention weights in the output of the encoder. Although this model was proposed for the machine translation task, Devlin et al. [35] used the transformer model to generate BERT word embeddings. Sun et al. [150] proposed a multi-tasking framework using transformers called ERNIE 2.0. In this framework, the model is continuously pretrained i.e., when a new task is presented the model is fine-tuned to accommodate the new task while retaining the previously gained knowledge. The model outperformed BERT. XLNet proposed by Yang et al. [162] used an autoregression model as opposed to the autoencoder model and outperformed BERT and ERNIE 2.0. A number of variations of BERT models were proposed based on the corpus used to train the model and by optimizing the computational resources. Lan et al. [60] proposed ALBERT, with two techniques to reduce the computational complexity of BERT namely 'factorized embedding parameterization' and 'cross-layer parameter sharing'. ALBERT outperformed all the above three models. Other variations of BERT models that use transformers include TinyBERT [54], RoBERTa [78, 133], and a domain-specific variation trained on a scientific corpus with a focus on the BioMedical domain the SciBERT [17]. Raffel et al. [121] proposed a transformer model with a well-defined corpus called 'Colossal Clean Crawled Corpus' or C4 to train the model named T5-11B. Unlike BERT they adopt a 'text-to-text framework' where the input sequence is attached with a token to identify the NLP task to be performed thus eliminating the two stages pre-training and fine-tuning. They propose five different versions of their model based on the number of trainable parameters each model has namely 1) T5-Small 2) T5-Base 3) T5-Large 4) T5-3B and 5)T511B and they have 60 million, 220 million, 770 million, 3 billion, and 11 billion parameters respectively. This model outperformed all other transformer-based models and achieved the state of the art results. As a result of their study, they confirm that the performance of the models increases with increased data and computational power and the performance can be further improved if larger models are built and it is important to note that in order to replicate their best model five GPUs are required among other resources. A compilation of the various transformer-based models and their Pearson's correlation on the STS-B dataset is provided below in Table 2.1.

Model Name	**Title**	**Year**	**Pearson's Correlation**
T5-11B	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer	2019	0.925
XLNet	XLNet: Generalized Autoregressive Pretraining for Language Understanding	2019	0.925
ALBERT	ALBERT: A Lite BERT for Self-supervised Learning of Language Representations	2019	0.925
RoBERTa	RoBERTa: A Robustly Optimized BERT Pretraining Approach	2019	0.922
ERNIE 2.0	ERNIE 2.0: A Continual Pre-training Framework for Language Understanding	2019	0.912
DistilBERT	DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter	2019	0.907
TinyBERT	TinyBERT: Distilling BERT for Natural Language Understanding	2019	0.799

Table 2.1: Pearson's Correlation of various transformer-based models on STS benchmark dataset.

Deep neural network-based methods outperform most of the traditional methods and the recent success of transformer-based models have served as a breakthrough in semantic similarity research. However, implementation of deep-learning models requires large computational resources, though variations of the models to minimize the computational resources are being proposed we see that the performance of the model takes a hit as well, for example, TinyBERT [54]. And the performance of the models is largely increased by the use of a bigger corpus which again poses the challenge of building an ideal corpus. Most deep-learning models are "black-box" models and it is difficult to ascertain the features based on which the performance is achieved, hence it becomes difficult to be interpreted unlike in the case of corpus-based methods that have a strong mathematical foundation. Various fields like finance, insurance, etc., that deal with sensitive data may be reluctant to deploy deep neural network-based methods due to their lack of interpretability.

2.5 Hybrid methods

Based on all the previously discussed methods we see that each has its advantages and disadvantages. The knowledge-based methods exploit the underlying ontologies to disambiguate synonyms, while corpus-based methods are versatile as they can be used across languages. Deep neural network-based systems, though computationally expensive, provide better results. However, many researchers have found ways to exploit the best of each method and build hybrid models to measure semantic similarity. In this section, we describe the methodologies used in some of the widely used hybrid models.

Types of hybrid semantic similarity methods:

- **Novel Approach to a Semantically-Aware Representation of Items (NASARI)** [24]: Camacho Collados et al. [24] proposed an approach the $NASARI$ where the knowledge source BabelNet is used to build a corpus based on which vector representation for concepts (words or group of words) are formed. Initially, the Wikipedia pages associated with a given concept, in this case, the *synset* of BabelNet, and all the outgoing links from the given page are used to form a sub-corpus for the specific concept. The sub-corpus is further expanded with the Wikipedia pages of the hypernyms and hyponyms of the concept in the BabelNet network. The entire Wikipedia is considered as the reference corpus. Two different types of vector representation were proposed. In the first method, weighted vectors were formed using lexical specificity. Lexical specificity is a statistical method of identifying the most representative words for a given text, based on the hypergeometric distribution (sampling without replacement). Let 'T and t' denote the total content words in the reference corpus RC and sub-corpus SC respectively and 'F and f' denote the frequency of the given word in the reference corpus RC and sub-corpus SC respectively, then lexical specificity can be represented by the below equation

$$spec(T, t, F, f) = -log_{10}P(X \geq f) \tag{2.13}$$

X represents a random variable that follows a hypergeometric relation with the parameters T, t and F and $P(X \geq f)$ is defined as,

$$P(X \geq f) = \sum_{i=f}^{F} P(X = i) \tag{2.14}$$

$P(X = i)$ is the probability of a given term appearing exactly i times in the given sub-corpus in hypergeometric distribution with T, t and F. The second method forms a cluster of words in the sub-corpus that share a common hypernym in the WordNet taxonomy which is embedded in BabelNet. The specificity is then measured based on the frequency of the hypernym and all its hyponyms in the taxonomy, even those that did not occur in the given sub-corpus. This clustering technique forms a unified representation of the words that preserve the semantic properties. The specificity values are added as weights in both methods to rank the terms in a given text. The first method of vector representation was called $NASARI_{lexical}$ and the second method was called $NASARI_{unified}$. The similarity between these vectors is calculated using the measure called Weighted Overlap [113] as,

$$WO(v_1, v_2) = \sqrt{\frac{\sum_{d \in O}(rank(d, \vec{v_1}) + rank(d, \vec{v_2}))^{-1}}{\sum_{i=1}^{|O|(2i)^{-1}}}} \tag{2.15}$$

where O denotes the overlapping terms in each vector and $rank(d, \vec{v_i})$ represent the rank of the term d in the vector v_i.

Camacho Collados et al. [25] proposed an extension to their previous work and proposed a third vector representation by mapping the lexical vector to the semantic space of word embeddings produced by complex word embedding techniques like *word2vec*. This representation was called as $NASARI_{embedded}$. The similarity is measured as the cosine similarity between these vectors. All three methods were tested across the gold standard datasets M&C, WS-Sim and SimLex-999. $NASARI_{lexical}$ achieved higher Pearson's and Spearman's correlation in average over the three datasets in comparison with other methods like ESA, *word2vec*, and *lin*.

- **Most Suitable Sense Annotation (MSSA)** [128]: Ruas et al. proposed three different methodologies to form word-sense embeddings. Given a corpus, the word-sense disambiguation step is performed using one of the three proposed methods: Most Suitable Sense Annotation (MSSA), Most Suitable Sense Annotation N Refined (MSSA-NR), and Most Suitable Sense Annotation Dijkstra (MSSA-D). Given a corpus each word in the corpus is associated with a *synset* in the WordNet ontology and *"gloss-average-vector"* is calculated for each *synset*. The gloss-average-vector is formed using the vector representation of the words in the gloss of each *synset*. MSSA calculates the gloss-average-vector using a small window of words and returns the *synset* of the word which has the highest gloss-average-vector value. MSSA-D, however, considers the entire document from the first word to the last word and then determines the associated *synset*. These two systems use Google News vectors[3] to form the synset-embeddings. MSSA-NR is an iterative model, where the first pass produces the synset-embeddings, that are fed back in the second pass as a replacement to gloss-average-vectors to produce more refined synset-embeddings. These synset-embeddings are then fed to a *word2vec* CBOW model to produce multi-sense word embeddings that are used to calculate the semantic similarity. This combination of MSSA variations and *word2vec* produced solid results in gold standard datasets like R&G, M&C, WS353-Sim, and SimLex-999 [128].

- **Unsupervised Ensemble Semantic Textual Similarity Methods (UESTS)** [45]: Hassan et al. proposed an ensemble semantic similarity method based on an underlying unsupervised word-aligner. The model calculates the semantic similarity as the weighted sum of four different semantic similarity measures between sentences S_1 and S_2 using the equation below

$$\begin{aligned} sim_{USETS}(S_1, S_2) = \alpha * sim_{WAL}(S_1, S_2) + \beta * sim_{SC}(S_1, S_2) \\ + \gamma * sim_{embed}(S_1, S_2) + \theta * sim_{ED}(S_1, S_2) \end{aligned} \tag{2.16}$$

$sim_{WAL}(S_1, S_2)$ calculates similarity using a synset-based word aligner. The similarity between text is measured based on the number of shared neighbors each term has in the BableNet taxonomy. $sim_{SC}(S_1, S_2)$ measures similarity using soft cardinality measure between the terms in comparison. The soft cardinality function treats each word as a set and the similarity between them as an intersection between the sets. $sim_{embed}(S_1, S_2)$ forms word vector representations using the word embeddings

[3]https://code.google.com/archive/p/word2vec/ .

proposed by Baroni et al. [15]. Then similarity is measured as the cosine value between the two vectors. $sim_{ED}(S_1, S_2)$ is a measure of dissimilarity between two given sentences. The edit distance is defined as the minimum number of edits it takes to convert one sentence to another. The edits may involve insertion, deletion, or substitution. $sim_{ED}(S_1, S_2)$ uses word-sense edit distance where word-senses are taken into consideration instead of actual words themselves. The hyperparameters α, β, γ, and θ were tuned to values between 0 and 0.5 for different STS benchmark datasets. The ensemble model outperformed the STS benchmark unsupervised models in the 2017 SemEval series on various STS benchmark datasets.

Hybrid methods exploit both the structural efficiency offered by knowledge-based methods and the versatility of corpus-based methods. Many studies have been conducted to build multi-sense embeddings in order to incorporate the actual meaning of words into word vectors. Iacobacci et al. formed word embeddings called "Sensembed" by using BabelNet to form a sense annotated corpus and then using *word2vec* to build word vectors thus having different vectors for different senses of the words. As we can see, hybrid models compensate for the shortcomings of one method by incorporating other methods. Hence the performance of hybrid methods is comparatively high. The first 5 places of SemEval 2017 semantic similarity tasks were awarded to ensemble models which clearly shows the shift in research towards hybrid models [27].

2.6 Conclusion

Measuring semantic similarity between two text snippets has been one of the most challenging tasks in the field of Natural Language Processing. Various methodologies have been proposed over the years to measure semantic similarity and this chapter discusses the evolution, advantages, and disadvantages of these methods. It is clear from the comparisons done in this chapter that each method has its advantages and disadvantages and it is difficult to choose the best mode. While the focus of recent research is shifted towards building more semantically aware word embeddings, and the transformer models have shown promising results, the need for determining a balance between computational efficiency and performance is still a work in progress. Research gaps can also be seen in areas such as building domain-specific word embeddings, addressing the need for an ideal corpus. This chapter would serve as a good foundation for researchers who intend to find new methods to measure semantic similarity.

Chapter 3

Domain Specific Complex Sentence Semantic Similarity Dataset

This chapter contains excerpts from the following article submitted to the journal - Experts system with applications:

- Chandrasekaran, D., & Mago, V. (2020). Domain Specific Complex Sentence (DCSC) Semantic Similarity Dataset. arXiv preprint arXiv:2010.12637.

A new benchmark dataset - the Domain Specific Complex Sentences (DSCS) dataset comprising of 50 sentence pairs with associated semantic similarity values provided by 15 human annotators is proposed. Readability analysis is performed to highlight the increase in complexity of the sentences in the existing benchmark datasets and those in the proposed dataset. Further, a comparative analysis of the performance of various word embeddings and the results justify the hypothesis that the performance of the word embeddings decrease with an increase in complexity of the sentences.

3.1 Introduction

One of the core components of Natural Language Processing (NLP) is assessing the semantic similarity between text data. The versatility of natural languages has made it a challenging task for researchers to capture the semantics of text data using numerical representations. Measuring the semantic of text data is essential in various NLP tasks like text summarization[94], topic modelling[77], text simplification[147], machine translation[160], question answering tasks[22], information retrieval[57] and so on. Extensive research has been carried out in the past decade in the field of semantic similarity to construct vector representations that preserve the syntactic and semantic properties of words. Word embeddings like *word2vec*[89] and *GloVe*[110] exploit the principle of the distributional hypothesis[42] "similar words occur together frequently". These methods use the advancements in machine learning techniques to capture the semantics of the words using large text corpora. Recent language models like BERT[35], RoBERTa[78], and ALBERT[60] use transformers to build vector representations of text data from underlying corpora by traversing through the corpora in both directions. Over the years various benchmark datasets have been used for comparing the performance of models in measuring semantic similarity between text data. Two of the most popular datasets are the STS benchmark dataset[139] and the SICK dataset[83] on which BERT models have achieved near-perfect results. Analyzing the readability of the sentences in these datasets, we find that the sentences in these datasets have a low readability index which is a measure of complexity of sentences. However, various real world applications of semantic similarity involves more complex sentences to be analysed[47]. We hypothesize that the performance of the existing word embedding models will decrease with the increase in the complexity of sentences. In this chapter, we investigate the sensitivity of the existing word embeddings to the complexity of sentences, by providing a new benchmark dataset, the Domain-specific complex sentence (DSCS) dataset with sentences of a higher degree of complexity than the existing datasets. The DSCS dataset comprises of 50 pairs of sentences from the computer science domain, with corresponding similarity scores provided by 15 human annotators. We determine the correlation of the similarity values obtained using existing word-embeddings with the human-annotated similarity values and the results clearly prove our hypothesis indicating a decrease in the performance of the models that achieved the state of the art results in the existing datasets. The remaining of the chapter is organised as follows. Section 3.2 of this paper provides a brief description of the existing research works carried out in the field of semantic similarity. Section 3 describes in detail two of the existing benchmark datasets and five different word-embeddings chosen for the comparative analysis. Section 3 discusses in detail the methodology adopted to construct the new benchmark dataset and provides a detailed description of the readability analysis that compares the complexity of sentences in the existing datasets to the sentences in the proposed dataset. Section 5 provides a comparative study of the performance of various word embeddings and provides an insight into the inferences made that would guide the future scope of this research.

3.2 Related work

Similarity between text data does not always attribute to the lexical or syntactic similarity between text. While two sentences that contain exactly the same words may mean completely different, it is possible that sentences with different sets of words provide the same meaning. Hence while assessing the similarity between text data, it is of importance to understand the meaning conveyed by the text. The similarity between the meaning of the text is known as semantic textual similarity (STS). For the past three decades, various semantic similarity methods have been proposed to measure semantic similarity. These methods are widely classified as knowledge-based methods [70, 80, 128, 130, 132] and corpus-based methods[67, 148]. The knowledge-based methods use structurally strong ontologies like Wordnet [91], DBPedia[19], Wikipedia[1], Wikitionary[2], etc. These ontologies are often used like graphs and various edge counting methods, consider the words in the taxonomy as nodes, and calculate the distance between the words using the edges between them. The greater the distance between the words the lower their similarity value[118]. However, these methods assume that the length of these edges to be similar which is not always the case. Another type of knowledge-based approach, called the feature-based methods, assess the similarity based on features of the words, like their dictionary definition, neighboring concepts, etc. derived from the underlying ontologies[132]. Knowledge-based methods are computationally simple and are efficient in distinguishing the different meanings of words solving the problem of ambiguity with concepts like polysemy and synonymy. However they are heavily dependent on the underlying taxonomies, they do not account for the versatility of natural language, and structured taxonomies for languages other than English are not common. Corpus-based semantic similarity methods use statistical principles to capture the contextual semantics of data using large underlying corpora. The principle of distributional hypothesis states that words with similar meanings occur together in documents and this principle forms the basis of most corpus-based methods, while these methods do not take into account the actual meaning of individual words. Word vectors, also called word embeddings, are constructed using corpus-based methods and the similarity is measured based on the angle or distance between these vectors. The dimensionality of these embeddings depends on the size of the corpus. While using significantly large corpora various dimensionality reduction

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

[2]https://www.wiktionary.org/

techniques like Singular Value Decomposition (SVD), Principal Component Analysis (PCA), and filtering techniques are used to achieve computational efficiency. These word embeddings are the fundamental components of recent techniques that use the advancements in deep neural networks to achieve a significant increase in performance in semantic similarity tasks. word2vec proposed by Milokov et al.[89] and GloVe vectors proposed by Pennington et al.[110] have proven to be major breakthroughs in the field of semantic similarity and they are two of the most widely used word embeddings to date. In 2019, Delvin et al.[35] proposed the Bidirectional Encoder Representation from Transformers (BERT) language model which used transformers to build word embeddings, which were further used for various downstream NLP applications. Variations of BERT models like, ALBERT[60] and RoBERTa[78] were also published in 2019 and have outperformed the existing semantic similarity models achieving state of the art results. Raffel et al.[122] proposed the T5: text-to-text transformer model which used the principle of transfer learning and was trained on a custom build corpus called "Colossal Clean Crawled Corpus" or C4. This model tied for the first place with ALBERT by achieving a Pearson's correlation of 0.925 on the STS dataset. In the following subsections, we describe in detail the two most widely used benchmark datasets for assessing the performance of semantic similarity methods, and five of the most popular word embeddings that we have chosen for comparison in this paper.

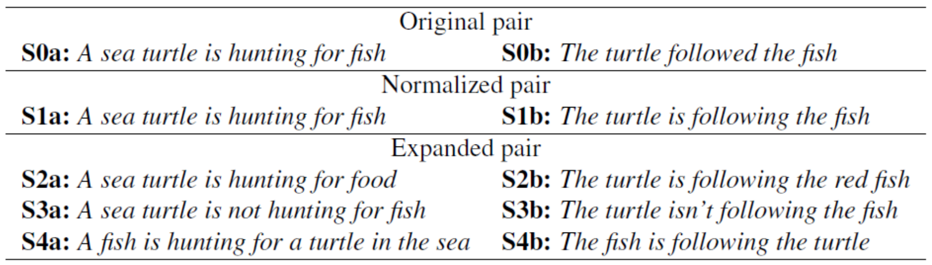
3.3 Datasets and Word embedding models

Semantic similarity Datasets

The first and the most widely used word-to-word similarity dataset, the R&G dataset[129], was proposed by Rubenstein and Goodenough in 1965 with 65 English noun pairs annotated by 51 native English speakers with similarity values ranging between 0 and 4. Some of the prominent datasets used are compiled in Table B.1 in Appendix B. While many datasets were published over the years as the benchmark for models measuring sentence level semantic similarity, the SICK dataset, and the STS datasets are those that gained significance. For our analyses and comparison, we choose these two datasets owing to their wide usage and popularity.

SICK Dataset[83]

Marelli et al.[83] compiled the SICK dataset for sentence level semantic similarity/relatedness in 2014 composed of 10,000 sentence pairs obtained from the ImageFlickr 8 and MSR-Video descriptions dataset. The sentence pairs were derived from image descriptions by various annotators. 750 random sentence pairs from the two datasets were selected, followed by three steps to obtain the final SICK dataset: sentence normalisation, sentence expansion and sentence pairing. Initially all the sentences were normalised by removing unwanted syntactic or semantic phenomena. This process was carried out by two different annotators and checked for compatibility. In instances of contradiction, a third annotator made the choice by analysing the two alternatives, and if both were correct a random choice was made. From the normalised sentences, 500 pairs were chosen for expansion. The process of expansion involved generating three different versions of the normalised sentence pairs - a similar sentence using meaning preserving transformation, a completely opposite version using negative transformation, and a sentence with same words but different meaning using random shuffling. Figure 3.1 shows an example as presented by the authors. Finally, each normalised sentence was paired with all its expanded version, along with some random pairing. A survey was then conducted on Amazon Mechanical Turk where the workers were requested to rank the similarity/relatedness over a scale of 1 to 5 with 1 representing that the sentences were highly dissimilar and 5 representing that the sentences were highly similar. Ten unique responses for each sentence pair were collected and the average of the ten ratings was assigned as the gold standard. Each sentence pair is associated with a relatedness score and a text entailment relation as well. The three entailment relations are "NEUTRAL, ENTAILMENT and CONTRADICTION."



Original pair	
S0a: *A sea turtle is hunting for fish*	**S0b:** *The turtle followed the fish*
Normalized pair	
S1a: *A sea turtle is hunting for fish*	**S1b:** *The turtle is following the fish*
Expanded pair	
S2a: *A sea turtle is hunting for food*	**S2b:** *The turtle is following the red fish*
S3a: *A sea turtle is not hunting for fish*	**S3b:** *The turtle isn't following the fish*
S4a: *A fish is hunting for a turtle in the sea*	**S4b:** *The fish is following the turtle*

Figure 3.1: Example of SICK dataset sentence expansion process[83].

STS Dataset [139]

In order to encourage research in the field of semantic similarity, semantic textual similarity tasks called SemEval have been conducted from 2012. The organizers of the SemEval tasks collected sentences from a wide variety of sources and compiled them to form a benchmark dataset against which the performance of the models submitted by the participants in the task was measured. While the dataset contains different tracks with sentences from different languages, we focus on the English component of the dataset. The English component of the dataset contains 8,295 sentence pairs of which 5,720 are provided as training samples and the remaining 2,575 sentences form the test set. The dataset is built over the years and contains the sentences that were used from 2012 to 2017. Table B.2 provides a breakdown of the source of the sentences in the dataset and the year they were appended to form the current version of the dataset. The sentences were annotated using Amazon Mechanical Turk. The quality of the annotators was assessed using the 'masters' provided by the platform and five unique annotations were obtained for each sentence pair. The similarity values ranged between the values 0 and 5, with 5 indicating that the sentences are completely similar and 0 indicating the sentences are completely dissimilar. The final similarity score was obtained by taking an average of the five unique responses from the annotators.

Word-embeddings

Distributed semantic vector representations of words called word embeddings have gained significant attention in the past decade, and a wide variety of word embeddings have been proposed over the years[23]. Word embeddings are constructed by analysing the distribution of words in any text data and are well known to capture the semantics of the words thus making them a significant component of a wide variety of semantic similarity algorithms. *word2vec* model uses neural networks to construct word embeddings and it has been one of the most widely used word-embeddings[23]. *GloVe* vectors employ word co-occurrence matrices to identify the distribution of words, which is then statistically used to build word vectors that capture the semantics of the target word with respect to its neighboring words. Pre-trained word embeddings provided by recent transformer based models achieved state of the art results in a wide range of NLP tasks, including semantic similarity. In this section we discuss in detail five of the popular word embeddings that are publicly available.

word2vec[89]

Mikolov et al. proposed a word embedding called word2vec 2013, using a simple neural network that converted the given input word to a dense vector representation. Two different models of word2vec were proposed namely the Skip-gram model and the Continuous Bag of Words (CBOW) model. In the skip-gram model, the neural network is optimized to predict a target word given its context words, whereas in the CBOW model, the neural network predicts the neighboring words given a target word. The value vector in the hidden layer of the optimized neural network is used as the vector representation of the word. The number of neurons in the hidden layer of the neural network determines the dimension of the word vector. The models are trained using Google News corpus, which contains 1 million words. The model produced state of the art results in 2013, and is used widely among researchers owing to the simplicity in construction of the neural network. The major claim of the authors was that when simple algebraic operations were performed on these vectors the results were closely related to human understanding. For example, the difference between V_k (vector representation for the word 'king') and V_m (vector representation for the word 'man') added to V_w (vector representation for the word 'women') provides a vector that is close to vector V_q (vector representation for the word 'queen'). This can be mathematically represented as,

$$Vq \simeq (V_k - V_m) + V_w \tag{3.1}$$

GloVe[110]

Researchers at Stanford University proposed a vector representation for words using the word to word co-occurrence matrices. Given a corpus, a global co-occurrence matrix is built where each row and column represents the words in the corpus. The underlying principle for the construction of these vectors is that similar words occur together. The model proposed uses 'log-bilinear regression' to create a word vector space with substructures that provide meaningful word-vector representations. GloVe vectors were trained on five different corpora like common web crawled corpus and the Wikipedia data dump resulting in 400,000 unique words to form the co-occurrence matrix. Pretrained word vectors of 3 different dimensions (50, 100 and 300) were released by the authors and they claimed that GloVe vectors outperformed word2vec and achieved the state of the art results in 2014.

BERT[35]

In 2019, the BERT transformer model surpassed the state of the art results in 11 different NLP tasks, including semantic similarity. BERT uses the transformer model proposed by Vaswani et al[157]. The BERT models follow two distinct steps to adapt to specific NLP tasks namely pretraining and fine-tuning. The transformer contains an encoder and decoder module, the encoder containing 6 identical layers stacked above each other. Each layer consists of sublayers comprising of a multi-head attention mechanism followed by a fully connected feed-forward neural network. The decoder is similar to the encoder, with one additional sub-layer of multi-head attention, which captures the attention weights in the output of the encoder. The model is pretrained using the Book corpus[165] and Wikipedia dump comprising of nearly 3300 million words. Pre-training is carried out with the help of two tasks namely 'Masked Language Model (MLM)' and 'Next Sentence Prediction (NSP)'. In the first task, random words in the corpus are masked and the model is optimized to predict the masked tokens. In the second task, the model is optimized to predict whether or not a sentence follows another given sentence. BERT models thus produce bidirectional representations of words taking into consideration the context of the word in both directions. In general, the BERT model is fine-tuned using labeled training data to accommodate a specific NLP task. The model was fine-tuned with STS dataset and achieved state of the art results.

RoBERTa[78]

Liu et al.[78] proposed a robustly optimized version of BERT, by replicating the work of Delvin et al.[35] and adding to it an improved training procedure. They added more training data and trained for a longer period of time achieving state of the results which proved the BERT architecture was equipped to perform better than many later models, but it was under trained. While BERT was trained on Book Corpus and Wikipedia corpus, RoBERTa model was trained on four different corpora namely Book Corpus, Common Crawl News dataset, OpenWebText dataset and the Stories dataset. RoBERTa uses variations of the pretraining tasks used by BERT model. It uses both static and dynamic masking, and by performing dynamic mask the training data is duplicated ten times thus enabling the model to encounter each masked tokens four times over the forty epoch training. The authors study the effect of 'Next Sentence Prediction' task by replacing it with prediction of subsequent segments or sentences in a document and prove that the performance increased by removing the NSP task. The model outperforms the BERT model and achieves state of the art results in 11 NLP tasks including semantic similarity.

ALBERT[60]

One of the major challenges in the BERT model is the time and resource requirement to pretrain a complex transformer model. Lan et al.[60] proposed a Lite version of BERT by employing two different parameter reduction techniques to aid in scaling the pre-trained models, namely Factorized Embedding Parameterization (FEP) and Cross-layer Parameter Sharing (CPS). Using FEP, the authors split the vector embedding matrix of the vocabulary into two smaller matrices thus making the size of the vector independent of the hidden layer of the model. Using CPS enables the sharing of parameters across layers thus preventing the increase in number of parameters as the depth of the layers increase. ALBERT also replaces one of the pretraining tasks - next sentence prediction in BERT with inter-sentence coherence loss that focuses on the coherence between two consecutive sentences. ALBERT has outperformed all the existing models and hold the highest Pearson's correlation in STS dataset.

3.4 Methodology

In this section, we discuss the methodology followed in building the proposed benchmark DSCS dataset. We follow three steps in the construction of the dataset namely 1) Selection of domain, 2) Selection of sentences 3) Annotation of similarity values. Owing to our familiarity and current research topic, we chose the domain of interest to be computer science. Our dataset comprises of 52 unique sentences, which are definitions of widely known topics in the field of computer science. The list of the topics chosen is shown in Table 3.1. In order to extract sentences with similar meanings we used three different sources namely the Wikipedia[3], Simple English Wikipedia[4] and the Merriam Webster Online dictionary[5]. The single sentence definitions of the chosen topics are selected from these sources resulting in fifty two unique sentences. Figure 3.2 shows an example of the definitions from various sources. In order to obtain dissimilar sentence pairs the sentences are paired among each other to form fifty final sentence pairs with no two sentences repeated more than twice.

[3] 1
[4] https://simple.wikipedia.org/wiki/Main_Page
[5] https://www.merriam-webster.com/

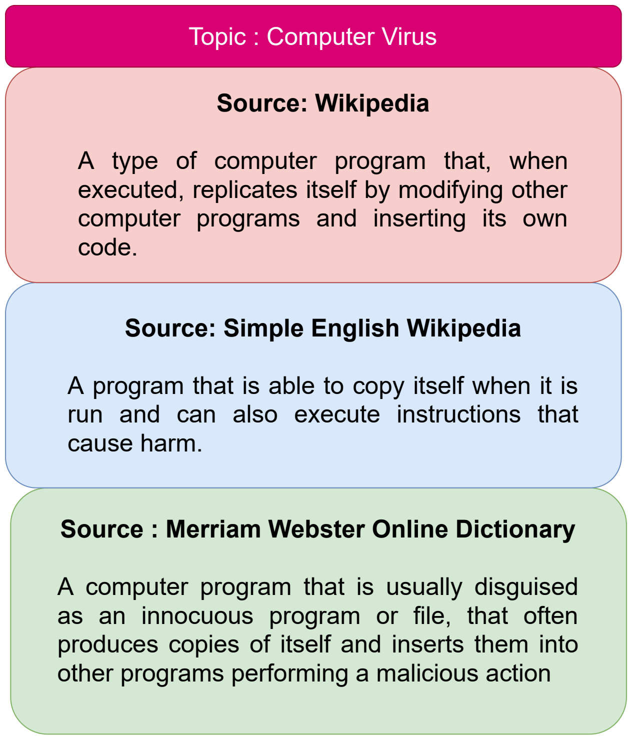

Topic : Computer Virus
Source: Wikipedia
A type of computer program that, when executed, replicates itself by modifying other computer programs and inserting its own code.
Source: Simple English Wikipedia
A program that is able to copy itself when it is run and can also execute instructions that cause harm.
Source : Merriam Webster Online Dictionary
A computer program that is usually disguised as an innocuous program or file, that often produces copies of itself and inserts them into other programs performing a malicious action

Figure 3.2: Example of Sentences in DSCS dataset.

Human Annotation

The sentence pairs were then marked with a binary annotation as "Similar" and "Dissimilar" based on whether they are the definition of the same topic. Then two separate surveys were conducted to obtain the human annotation of the similarity value between the sentence pairs. The initial survey was conducted among five graduate students pursuing a Masters in Computer Science. The students were requested to mark the similarity between provided sentence pairs over a range of values between 0 and 5, where 0 indicates that the sentences are completely dissimilar and 5 indicates that the sentences are completely similar. The survey was then extended to Amazon Mechanical Turk (MTurk) requesting 10 unique responses for each sentence pair. The survey in MTurk was restricted to North America for language

Table 3.1: Topics used to build DSCS dataset.

SNo	Topic	SNo	Topic
1	Computer science	10	Psuedo code
2	Computer program	11	Programming language
3	Algorithm	12	Data analytics
4	Data structures	13	Computer Security
5	Artificial Intelligence	14	Computer virus
6	Computer programming	15	Cloud computing
7	Operating systems	16	Server
8	Database	17	Firewall
9	Computer Architecture	18	Outlier.

expertise and the Turker was required to have US graduate level of expertise for domain expertise. Specific instructions were provided to the workers that sufficient domain knowledge is required to participate in the survey. However, since the actual qualification of the user cannot be determined or restricted in MTurk the responses of the workers were collected and examined for grave irregularities. The responses of all the workers were compared to the existing binary annotation of the dataset. If more than 80% of the similarity values provided by a worker contradicted the binary annotation then the users' responses were removed. By repeating this process 10 unique responses for 50 sentence pairs were obtained. The results of the two surveys were combined to obtain 15 unique values for each sentence pair and the similarity score was calculated using the weighted average of the 15 unique responses using the formula below.

$$S = \frac{\sum_{i=0}^{i=5}(w_i \times s_i)}{\sum_{i=0}^{i=5} w_i} \tag{3.2}$$

where,

s_i represent the values from 0 to 5 respectively and

w_i represent the corresponding weights and are calculated as,

$$w_i = \left(\frac{\text{No of responses with } i \text{ similarity score}}{\text{Total number of responses}} \right) \tag{3.3}$$

Readability Analysis

Readability indices are used by researchers to measure the complexity of text data mostly in text simplification tasks [140],[7]. In order to prove that the sentences chosen for building this dataset are more complex than the existing gold standard datasets a comparative readability analyses is conducted between the two existing benchmark datasets and the proposed dataset.The below-mentioned readability grade-level scores indicate the grade level of education required by the reader to comprehend the given text which in turns reflects the complexity of the sentences. For example, a readability index of 10.4 indicates that a student of grade 10 would be able to read the given text. Various readability indices used and the formula for determining the scores are provided below.

- **Flesch-Kincaid Grade Level[59]**

$$= 0.39 \left(\frac{totalwords}{totalsentences} \right) + 11.8 \left(\frac{totalsyllables}{totalwords} \right) - 15.59 \tag{3.4}$$

- **Coleman-Liau Index[31]**

$$= 0.588L - 0.296S - 15.8 \tag{3.5}$$

where,

L denotes the number of characters per 100 words and

S denotes the number of sentences per 100 words.

- **Automated readability Index[59]**

$$= 4.71 \left(\frac{characters}{words} \right) + 0.5 \left(\frac{words}{sentences} \right) - 21.43 \tag{3.6}$$

- **Linsear Write**

  For each sample of 100 words,

  $$r = \frac{1 * (Easywords) + 3 * (Hardwords)}{No\ of\ sentences\ in\ sample}$$

  where,

  Easywords = words with less than 2 syllables and

  Hardwords = words with more than 3 syllables.

  $$LW = \begin{cases} \left(\frac{r}{2}\right), & \text{if } r > 20 \\ \left(\frac{r}{2} - 1\right) & \text{if } r \leq 20 \end{cases} \tag{3.7}$$

- **Gunning fog index**[43]

  $$= 0.4\left[\left(\frac{words}{sentences}\right) + 100\left(\frac{complex\ words}{words}\right)\right] \tag{3.8}$$

  where,

  complex words = words consisting more than or equal to three syllables

- **Text Standard**:

  An aggregated score based on all the above readability indices.

The STS training dataset contains 10,818 unique sentences and the SICK dataset contains 6,076 unique sentences. On analysing the complexity of these sentences using the above mentioned readability indices we find that 70% of sentences in STS dataset and 90% of sentences in SICK dataset have a aggregate readability score below 10, while only 25% of the sentences in the proposed dataset are below the index 10. This clearly indicates that the two existing datasets have predominantly simpler sentences. In order compare the complexity of the datasets, we select the readability indices of 52 random sentences

Table 3.2: Configuration of BERT, RoBERT, and ALBERT models compared in this paper.

Model Name	**Trans former Blocks**	**Hidden Layers**	**Attention heads**	**Total Parameters**
BERT$_{\text{BASE}}$	12	768	12	110M
BERT$_{\text{LARGE}}$	24	1024	16	340M
RoBERTa$_{\text{BASE}}$ and RoBERTa$_{\text{LARGE}}$	24	1024	16	340M
ALBERT$_{\text{XLARGE}}$	24	2048	32	60M
ALBERT$_{\text{XXLARGE}}$	12	4096	64	235M

from the the two benchmark datasets and 52 sentences from DSCS dataset. This process is repeated with three different seeds for random selection. Figure 3.3 shows the results of the comparison between the mean value of six different readability indices among the three datasets repeated thrice and we can clearly see a significant increase in the complexity of the sentences. The results show us that while sentences in STS dataset and SICK dataset can be interpreted by 6th graders and 4th graders respectively, DSCS requires the knowledge of a 12th grader to comprehend the meaning of the provided sentences clearly indicating the increase in complexity in the sentences of the proposed dataset.

3.5 Comparative analysis

Experimental setup

We perform a comparative analysis between the five chosen word embeddings, to assess the sensitivity of the word embeddings to the complexity of the sentences, by comparing their performance across the two benchmark datasets and the proposed dataset. Since this research experiment focuses on the sensitivity of the word embeddings themselves, the sentence vectors are formed by simply taking the mean of the word vectors. In transformer based models a mean pooling is added to the model to form the sentence vectors. Initially we replicate the results provided by the authors of each model using the STS dataset, and use the same models to measure the similarity scores for both SICK dataset and the proposed DSCS dataset. In the first model, the pre-trained *word2vec* model, trained on the Google News dataset containing 300 dimensional vectors with a vocabulary of 3M words provided by *gensim* python library is used for building the word vectors. Then, the 300 dimensional *GloVe* vectors pre-trained on a Wikipedia dump corpus with 6B tokens provided by the *gensim* library is used to build the word vectors. However, since the proposed dataset is from the computer science domain, we use a specific corpus - the computer science corpus to train the word2vec and GloVe models initialized with the pretrained weights using transfer learning. The PetScan[6] tool that traverses through the Wikipedia categories and subcategories based on provided keywords is used for building the corpus. Using 'computer_science' as the category and a depth '2' which indicates the level of subcategories to be included, the 'computer science corpus' containing 4M tokens is used to train both the word2vec and GloVe models. Since the different BERT models do not provide explicit word embeddings, we use Sentence-BERT framework proposed by Reimer's et al.[123] to compare their performance. The various BERT models selected for comparison and their configuration is listed above in Table 3.2. We use the SentenceTransformer python library to initialize the model with the weights of pretrained BERT models followed by a mean pooling layer to form the sentence vectors. In order to factor in the effect of 'fine-tuning' one of the prominent characteristic of transformer based models we fine tune the BERT models, with the AllNLI dataset. To estimate the impact of supervised learning in the quality of the word vectors provided by transformer based models we experiment with BERT models fine tuned with both AllNLI

[6]https://petscan.wmflabs.org/

Table 3.3: Pearson's correlation and Spearman's correlation percentages of the word embedding models in STS Benchmark Dataset, SICK dataset, and DSCS dataset.

Supervision	**Model**	**Pearson's Correlation**			**Spearman's Correlation**		
		STS	**SICK**	**DSCS**	**STS**	**SICK**	**DSCS**
Unsupervised	word2vec	62.61	72.67	48.30	58.71	62.13	49.04
	GloVe	45.42	63.53	43.90	46.88	55.59	50.14
	word2vec + CS Corpus	56.77	67.75	**52.35**	53.27	59.38	46.79
	GloVe + CS Corpus	41.09	60.44	48.02	42.85	54.20	42.52
Partially supervised	BERT large + NLI + Mean Pooling	76.17	73.65	54.78	79.19	73.72	54.47
	BERT base + NLI + Mean Pooling	74.11	72.95	50.91	76.97	72.89	47.73
	RoBERTa large + NLI + Mean Pooling	76.26	74.35	54.54	78.69	74.01	46.36
	RoBERTa base + NLI + Mean Pooling	74.58	76.05	37.74	77.09	74.44	39.48
	ALBERT xxlarge + NLI + Mean Pooling	78.56	77.78	45.44	79.55	75.37	41.33
	ALBERT xlarge + NLI + Mean Pooling	77.68	73.90	**54.90**	80.12	74.78	57.68
Supervised	BERT large + NLI + STSB+ Mean Pooling	84.63	80.97	64.69	85.25	78.48	60.48
	BERT base + NLI + STSB+ Mean Pooling	84.18	82.16	67.79	85.04	78.43	64.02
	RoBERTa large + NLI + STSB+ Mean Pooling	85.54	82.32	71.98	86.42	78.40	66.78
	RoBERTa base + NLI + STSB+ Mean Pooling	84.26	81.78	50.57	85.26	77.43	46.46
	ALBERT xxlarge + NLI + STSB + Mean Pooling	92.58	85.33	**73.12**	90.22	80.03	67.54
	ALBERT xlarge + NLI + STSB + Mean Pooling	91.59	83.28	71.45	90.73	81.52	66.97

dataset and the STS dataset on all three datasets included in the comparison. While the T5 model proposed by Raffel et al.[122] is one of the models that achieved the best performance in the STS dataset, due to computational constraints we were not able to replicate the model hence it is not included in our comparison.

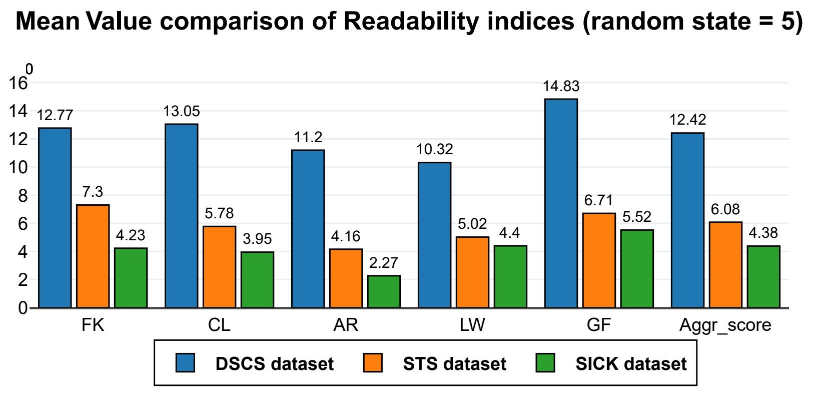
Results and Discussion

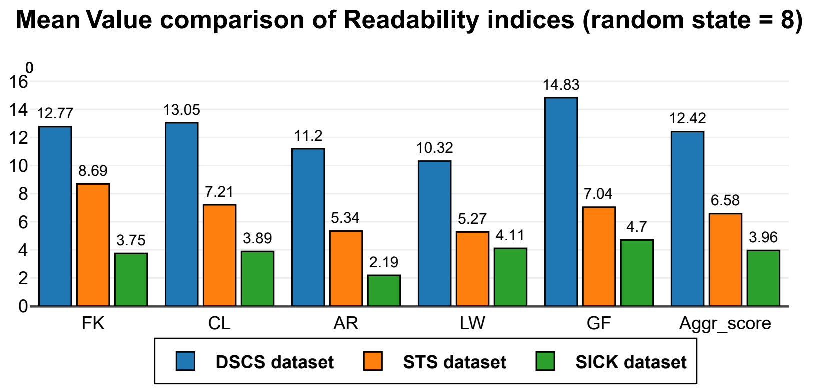
The correlation between the similarity scores provided by human annotators and the similarity scores calculated by the models is used as the measure to estimate the performance of the word embedding models. Since both Pearson's correlation and Spearman's correlation are used by the authors of the models chosen for comparison, we depict our results using both the correlations. The results are categorized as three different sections based on the level of supervision used for the models. The first section records the performance of the unsupervised word embedding models, the word2vec and GloVe and their variations trained on the 'computer science corpus'. The second section shows the performance of the two variants of BERT, RoBERTa and ALBERT models, fine tuned using AllNLI dataset, we consider this as partial supervision since the text entailment tasks are similar to semantic similarity tasks, but the model has not been trained on sentences from any of the three datasets used for comparison. The third section comprises the results of the same models trained (fine-tuned) using both AllNLI and the STS dataset. The results are provided in Table 3.3.

The performance of the word-embeddings decrease considerably when tested on the proposed dataset, which proves our initial hypothesis that performance decreases with increase in complexity. While ALBERT-xxlarge model achieves the best performance in the proposed dataset with 73.12% Pearson's Correlation and 67.54% Spearman's correlation, we can see that these results are sub-par in comparison to the 92.58% correlation achieved in the existing benchmark dataset. It is important to note that, though the sentences in the proposed dataset are definitions of topics from a particular domain, they are derived from sources that are commonly used by everyone. Hence these sentences are comparatively simpler than the sentences that we might encounter in scientific or academic articles. Hence, we confer that there is an imminent need to explore venues to improve the quality of these word embeddings to capture the semantics in complex documents. Next, we see that the correlation in the proposed dataset increased in word2vec and GloVe models trained using a custom built domain-specific corpus than the models trained on general corpora indicating the need for custom built efficient corpora for domain specific tasks. We also see that the BERT models fine-tuned to a specific NLP task like semantic similarity do not perform as effectively in the proposed dataset indicating the impact of the training data used in the fine-tuning process. With the insights we have gathered through the experiments we described in this chapter, we propose to focus our future research on two aspects. We intend to repeat the process and add more number of sentences to the dataset and accommodate more domains thus building a large complex sentence dataset which can in turn be used by transformer-based models for fine tuning. Secondly, we intend to focus on building more complex domain specific corpora compared to the common crawl dataset which predominantly contains simpler text, thus training word embedding models to capture semantics in more complex sentences.

3.6 Conclusion

Measuring semantic similarity between text data has been one of the most challenging tasks in the field of Natural Language Processing. Various word embedding models have been proposed over the years to capture the semantics of the words in numeric representations. In this paper we propose a new benchmark dataset, that contains sentences that are more complex than the existing dataset and measure the sensitivity of the existing word embedding models to the increase in complexity of the sentences. On performing a comparative analysis between five different word embedding models we prove our hypothesis that the performance of word embeddings models in semantic similarity tasks decreases with increase in complexity.





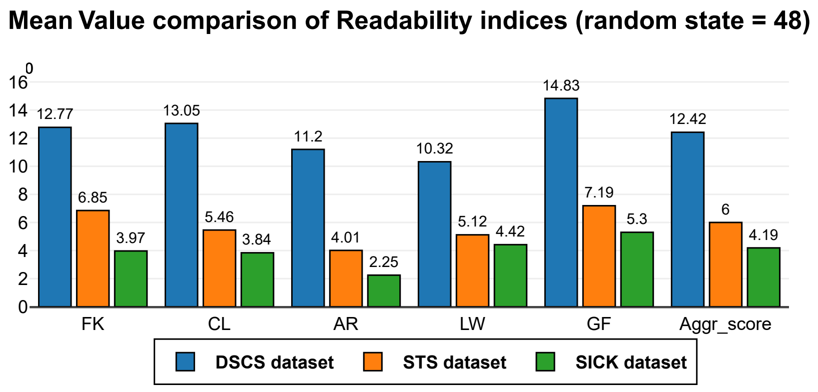


Figure 3.3: Comparison of mean readability indices among three given datasets.

Chapter 4

Automating Transfer Credit Assessment in Student Mobility - A Natural Langugage Processing-based Approach.

This chapter contains excerpts from the article submitted to a journal:

- Chandrasekaran D. & Mago, V. (2021). Automating Transfer Credit Assessment in Student Mobility - A Natural Langugage Processing-based approach.

Using the advancements in the field of Natural Language Processing, this chapter proposes a model to automate the process of transfer credit assessment. The chapter explores the three important components of the model namely the semantic similarity module, taxonomic similarity module and the aggregation module.

4.1 Introduction

With the significant growth in the enrollment of students in post-secondary institutions and the growing trend of interest in diversifying one's scope of education, there is an increasing demand among the academic community to standardize the process of student mobility. Student mobility is defined as "any academic mobility which takes place within a student's program of study in post-secondary education [55]." Student mobility could be both international and domestic. While there are various barriers to student mobility, offering transfer credits for students moving from one post-secondary institution to another is considered one of the most critical and labor-intensive tasks [47]. Various rules and regulations are proposed and adopted by institutions across different levels (provincial, federal, or international) to assess transfer credits, but most of these methods are time-consuming, subjective, and influenced by undue human bias. The key parameter used in assessing the similarity between programs or courses across institutions is learning outcome (LO), which provides context on the competencies; achieved by students on completion of a respective course or program. To standardize this assessment, LOs are categorized into various levels based on Bloom's taxonomy. Bloom's taxonomy proposed by Benjamin Bloom [20] attempts to classify the learning outcomes into six different categories based on their "complexity and specificity", namely *knowledge, comprehension, application, analysis, synthesis, and evaluation.*

Semantic similarity, being one of the most researched Natural language processing (NLP) tasks, has seen significant breakthroughs in recent years with the introduction of transformer-based language models. These language models employ attention mechanisms to capture the semantic and contextual meaning of text data and represent them as real-valued vectors, that are aligned in an embedding space such that the angle between these vectors provides the similarity between the text in consideration. In an attempt to reduce the inherent complexity and bias, and exploit the advancements in the field of NLP, we propose an algorithm that determines the similarity between courses; based on the semantic and taxonomic similarity of their learning outcomes. The proposed algorithm

- ascertains taxonomic similarity of LOs based on Bloom's taxonomy.
- determines semantic similarity of LOs using RoBERTa language model.
- provides a flexible aggregation method to determine the overall similarity between courses.

4.2 Background

This section provides a brief overview of the student mobility process across the world and the structural organization of learning outcomes. Various semantic similarity methods developed over the years are discussed in the final sub-section, thus providing an insight into the necessary concepts to understand the importance of the research and the choices made to develop the proposed methodology.

Student Mobility

The movement of students across institutions for higher education has existence for decades in the form of international student exchange programs, lateral transfers, and so on. Governments across the world follow different measures to standardize the process to ensure transparency and equity for students. According to the Organisation for Economic Co-operation and Development (OCED) indicators, there are approximately 5.3 million internationally mobile students [105]. Mobile students include both international students who cross borders to pursue education and foreign students who are from a different nationality than the country of the host institution. Mobile students face a wide range of barriers both academic and non-academic. Academic barriers include the lack of necessary qualifications and non-transferability of credits, while non-academic barriers may include social, cultural, financial, and psychological barriers. Governments across the world have taken various measures to reduce these barriers to enable a smooth transition for students. The Bologna process formed as a result of the Bologna declaration of 1999, provides guidance for the European Higher Education Area comprising 48 countries in the standardization of higher education and credit evaluation. In the United Kingdom, institutions like Southern England Consortium for Credit Accumulation and Transfer (SEEC) and Northern Universities Consortium for Credit Accumulation and Transfer (NUCCAT) oversee the collaboration between universities to allocate academic credits which are treated as currency awarded to students on completion of requirements. Canada offers provincial supervision of articulation agreements between institutions with the provinces of British Columbia and Alberta leading and the provinces of Ontario and Saskatchewan following yet way behind. The Ontario Council on Articulation and Transfer (ONCAT) carries out funded research to explore venues to increase the agreements between universities and colleges in Ontario. The credit transfer system in the United States is decentralized and often carried out by non-profit organizations designated for this specific purpose. In Australia, the eight most prominent universities established the Go8 credit transfer agreement to offer credit to students who move between these institutions. While there are various such governances on a national level, most international credit evaluations are carried out in a need-based manner. In addition to being an academic barrier; credit evaluation also has a direct impact on one of the most important non-academic barriers - the financial barrier. Hence, all agencies offer special attention to make this process fair and accessible to the population of mobile students worldwide.

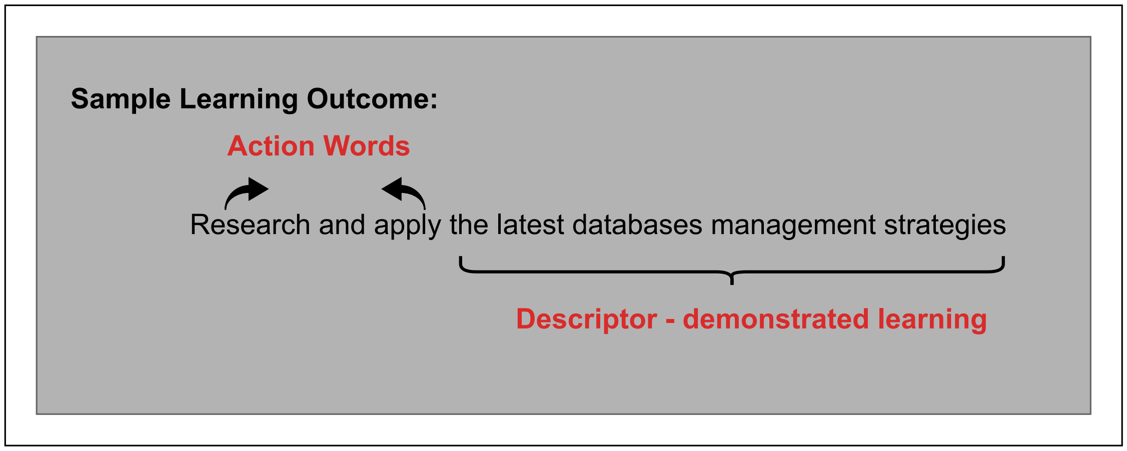

Sample Learning Outcome:
Action Words
Research and apply the latest databases management strategies
Descriptor - demonstrated learning

Figure 4.1: An example of a Learning Outcome.

Learning Outcomes

Credit evaluation is carried out by domain experts in the receiving institution by analyzing the learning outcomes of the courses the students have completed in their previous institution. Learning outcomes are often statements with two distinct components namely the action words and the descriptor. The descriptor part provides the knowledge the students have learned in a given course or program and the action words provide the level of competency achieved for each specific knowledge item. An example of learning outcomes for a computer programming course is provided in Figure 4.1. The taxonomy for educational objectives was developed by Bloom et al. [20] and later revised by Anderson et al. [11]. The six levels of the original taxonomy are *knowledge, comprehension, application, analysis, synthesis, and evaluation.* In the revised version, two of these levels were interchanged and three levels were renamed to provide better context to the level of the acquired knowledge. Hence the levels of the revised taxonomy are *"Remember, Understand, Apply, Analyze, Evaluate, and Create."* Each level encompasses sub-levels of more concrete knowledge items and these are provided in Figure 4.2. In order to estimate the similarity between learning outcomes, it is important to understand their structure and organization.

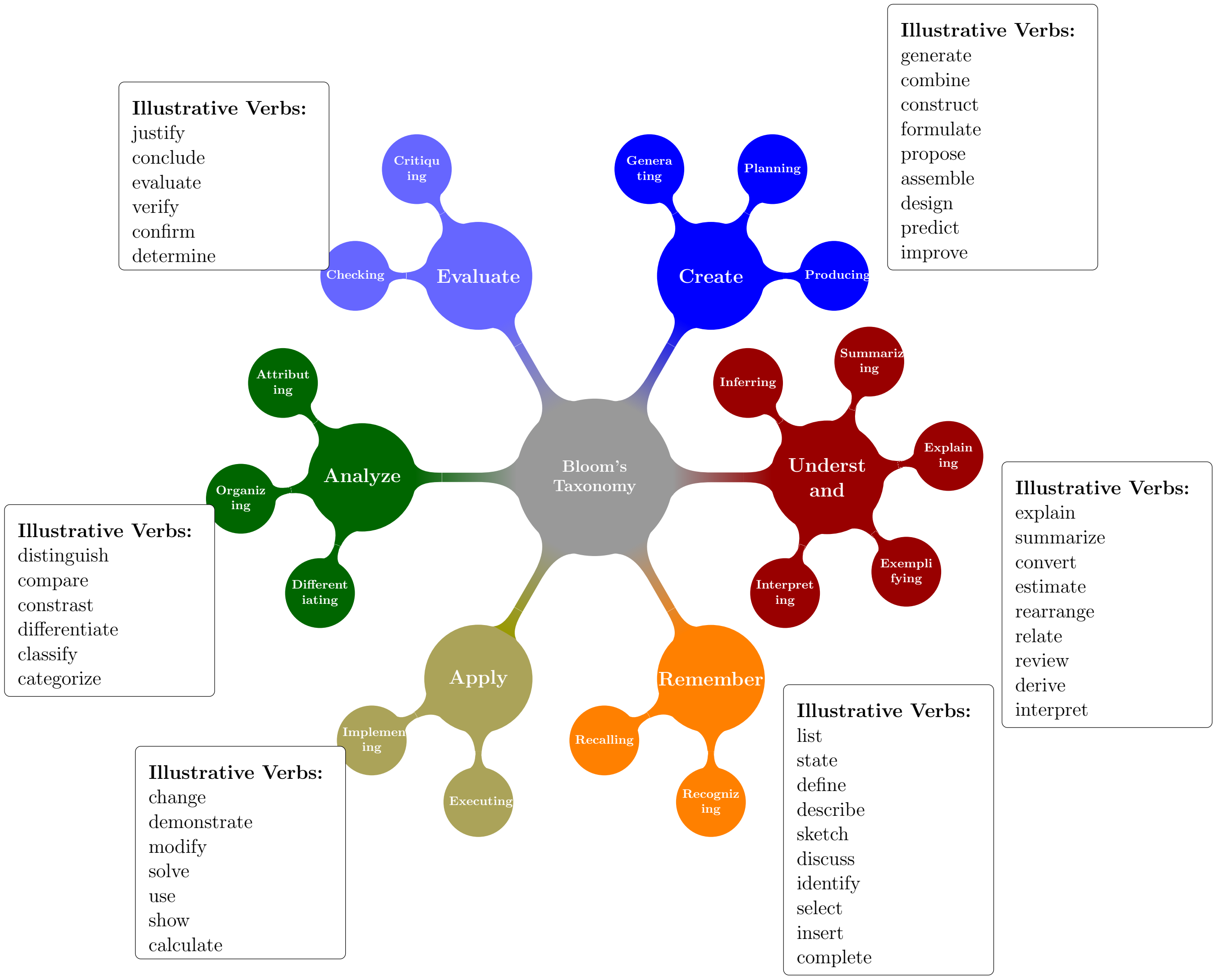


Figure 4.2: Bloom's taxonomy and corresponding illustrative verbs.

Semantic Similarity

The semantic textual similarity (STS) is defined as the similarity in the meaning of text data in consideration. Various semantic similarity methods proposed over the past two decades can be broadly classified as knowledge-based and corpus-based methods [29]. Knowledge-based methods rely on ontologies like WordNet, DBPedia, BabelNet, etc. The ontologies are conceptualized as graphs where the words represent the nodes grouped hierarchically, and the edges represent the semantic relationship between the words. Rada et al. [118] followed a straightforward approach and introduced the *path* measure in which the semantic similarity is inversely proportional to the number of edges between the two words. However, this method ignored the taxonomical information offered by the underlying ontologies. Wu et al., [161] proposed the *wup* measure that measured the semantic similarity in terms of the least common subsumer (LCS). Given two words, LCS is defined as the common parent they share in the taxonomy. Leacock et al. [68] proposed the *lch* measure by extending the *path* measure to incorporate the depth of the taxonomy to calculate semantic similarity. The formulations of these methods are provided in Section 2.2 . Other knowledge-based approaches include feature-based semantic similarity methods, that calculate similarity using the features of the words like their dictionary definition, grammatical position, etc., and information content-based methods that measure semantic similarity using the level of information conveyed by the words when appearing in a context. Corpus-based semantic similarity methods construct numerical representations of data called embeddings using large text corpora. Traditional methods like Bag of Words (BoW), Term Frequency - Inverse Document Frequency (TF-IDF) used one-hot encoding techniques or word counts to generate embeddings. These methods ignored the polysemy of text data and suffered due to data sparsity. Mikolov et al. [89] used a simple neural network with one hidden layer to generate word-embeddings when used in simple mathematical formulations, produced results that were closely related to human understanding. Pennington et al. [110] used word co-occurrence matrices and dimension reduction techniques like PCA to generate embeddings. The introduction of transformer-based language models, which produced state-of-the-art results in a wide range of NLP tasks, resulted in a breakthrough in semantic similarity analysis as well. Vaswani et al. [157] proposed the transformer architecture, which used stacks of encoders and decoders with multiple attention heads for machine translation tasks. Delvin et al. [35] used this architecture to introduce the first transformer-based language model, the Bidirectional Encoder Representations from Transformers (BERT) that generated contextualized word embeddings. BERT models were pre-trained on large text corpora and further fine-tuned to a specific task. Various versions of BERT were subsequently released namely, RoBERTa [78] - trained on a larger corpus over longer periods of time, ALBERT [60] - a lite version achieved using parameter reduction techniques, BioBERT [69] - trained on a corpus of biomedical text, SciBERT [17] - trained on a corpus of scientific articles, and TweetBERT [117] - trained on a corpus of tweets. Other transformer-based language models like T5 [121], GPT [119], GPT-2 [120], etc., use the same transformer architecture with significantly larger corpora and an increased number of parameters. Though these models achieve state-of-the-art results the computational requirements render them challenging to implement in real-time tasks [146].

4.3 Methodology

This section describes in detail the three modules of the proposed methodology namely,

- Pass 1: Taxonomic similarity
- Pass 2: Semantic similarity
- Pass 3: Aggregation

Given the learning outcomes of the courses in comparison, Pass 1 generates a *taxonomic_similarity_grid*, and Pass 2 generates a *semantic_similarity_grid* where the rows and columns are populated with the learning outcomes, and the cells are populated with the taxonomic similarity value and the semantic similarity value. These two grids are further passed on to Pass 3 where the final similarity between learning outcomes is assessed factoring in both the similarity values and further aggregated to arrive at the course level similarity. The architecture of the proposed model is presented in Figure 4.3.

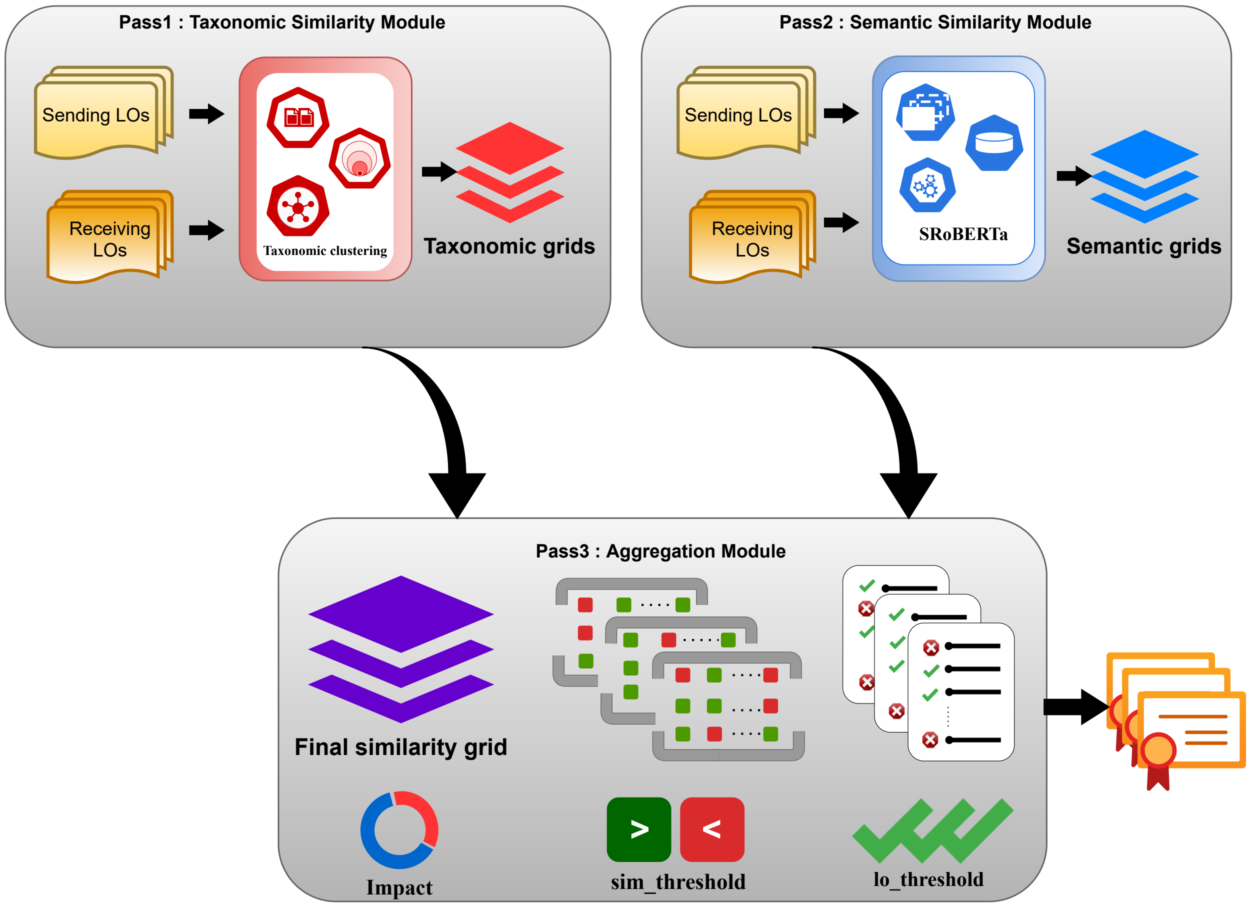


Figure 4.3: Proposed model architecture.

Pass 1: Taxonomic similarity

A clustering-inspired methodology is proposed to determine the taxonomic similarity between learning outcomes. Six different clusters, one for each level in Bloom's taxonomy are initialized with a list of illustrative verbs that best describe the cognitive competency achieved, specifically in the field of engineering [151], as shown in Figure 4.2. In this pass, the verbs in the learning outcomes are identified using spaCy pos tagger[1] and Wordnet synsets [91] and then these verbs are used to determine the cluster to which the learning outcomes are most aligned with. While encountering verbs already available in the list, a straightforward approach is followed and the learning outcomes are assigned to the respective cluster. However, for learning outcomes with new verbs, an optimal cluster is determined based on the semantic similarity between the new verb and the verbs in the existing clusters. The best measure to calculate this similarity is determined as a result of the comparative analysis carried out between various knowledge-based and corpus-based semantic similarity measures. Three knowledge-based measures namely *wup*, *lch*, and *path* are measured using Worndet ontology. In this ontology, there are more than one synsets for verbs hence it is necessary to identify the best synset. Given a verb pair, *wup*, *lch*, and *path* select the first synset of the verbs, whereas *wup_max*, *lch_max*, *path_max* identifies the synset that has the maximum similarity with the verb pairs. Gerz et al. [40] proposed SimVerb-3500 a benchmark dataset consisting of verb pairs with associated similarity values provided by annotators using crowd-sourcing techniques. The performance of six knowledge-based semantic similarity measures and word embeddings models (word2vec and Glove) on the SimVerb-3500 is compared and the results are depicted in Fig. 4.4. The best results are achieved by *wup_max* [161], hence used in the clustering process. Silhouette width is defined as "the measure of how much

[1]https://spacy.io/usage/v3

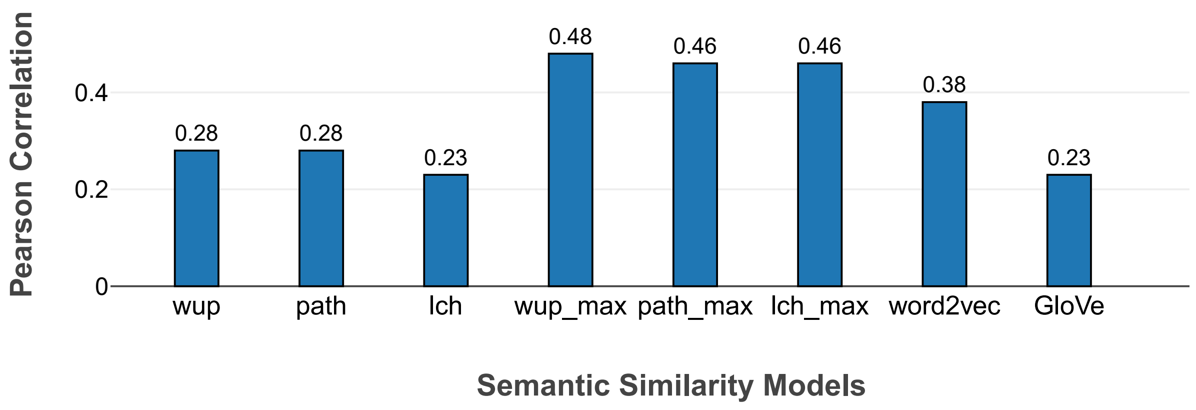


Figure 4.4: Pearson's correlation of various semantic similarity measures on Simverb3500 dataset.

more similar a data point is to the points in its own cluster than to points in a neighboring cluster[127]". The silhouette width of a verb $\mathcal{V}_i$ is measured as,

$$\mathcal{S}_i = \frac{b_i - a_i}{max(a_i, b_i)} \tag{4.1}$$

where, a_i is the average distance between the verb and the verbs in its own cluster and b_i is the average distance between the verb and the verbs in its neighboring cluster. The following equations formulate the calculation of these distance measures.

$$a_i = \frac{\sum\limits_{\mathcal{V}_j \in C(\mathcal{V}_i)} d(\mathcal{V}_i, \mathcal{V}_j)}{|C(\mathcal{V}_j)|} \qquad b_i = \frac{\sum\limits_{\mathcal{V}_k \in C_{neig}(\mathcal{V}_i)} d(\mathcal{V}_i, \mathcal{V}_k)}{|C_{neig}(\mathcal{V}_k)|}$$

$\mathcal{V}_j$ represents the verbs in the same cluster C and $\mathcal{V}_k$ represenst the verbs in the neighboring cluster C_{neig}, the function $d(\mathcal{V}_i, \mathcal{V}_j)$ calculates the distance between the verbs in this case the *wup_measure*. The value of $\mathcal{S}$ ranges from -1 to 1 where values closer to 1 indicate that the data point is assigned to the correct cluster. Based on this principle, when the model identifies a *new_verb* in a learning outcome, the silhouette width for each cluster is determined and the learning outcome is assigned to the cluster with maximum silhouette width. For learning outcomes with more than one verb, the verb assigned to the highest taxonomic level determines the final cluster value of the learning outcome in question. Since each cluster represents a corresponding level of competency in Bloom's taxonomy, the final taxonomic similarity between the learning outcomes is measured as,

$$taxonomic\_similarity\ (lo1,\ lo2) = abs\ (C_{lo1} - C_{lo2}) \tag{4.2}$$

, where C_{lo1} and C_{lo2} represent the cluster IDs of the learning outcomes in comparison calculated as,

$$C_{loi} = max\ \left(C(v_1), C(v_2)....C(v_n)\right) \tag{4.3}$$

Where n is the number of verbs in the learning outcome loi and $C(v_n)$ represents the respective cluster of the verb v_n. For two courses having m and n number of learning outcomes, a $m \times n$ dimensional *taxonomic_similarity_grid* is constructed and populated with the respective *taxonomic_similarity* values.

Pass 2: Semantic Similarity

Recent transformer-based language models generate contextual word embeddings that are fine-tuned for a specific NLP task, and various researchers have attempted to generate sentence embeddings by averaging the output from the final layer of the language models [163]. However, Reimers et al. [123] established that these techniques yielded poor results in regression and clustering tasks like semantic similarity and proposed the Sentence BERT (SBERT) model to address this issue. SBERT is a modified version of BERT-based language models where a siamese network and triple network structures are added to the final layer of the BERT network to generate sentence embeddings that capture the semantic properties and thus can be compared using cosine similarity. Reimers et al. [123] compared the performance of both SBERT and SRoBERTa language models in the STS [139] and SICK [82] benchmark datasets. Owing to the fact that the model is specifically designed for semantic similarity tasks, the computational efficiency of the model architecture, and the significant performance achieved, SRoBERTa is used in this pass to measure the semantic similarity of the LOs. SRoBERTa uses the base RoBERTa-large model with 24 transformer blocks, 1024 hidden layers, 16 attention heads, and 340M trainable parameters with a final mean pooling layer. The model is trained on the AllNLI dataset which contains 1 million sentence pairs categorized into three classes namely 'contradiction, entailment, and neutral', and the training data of the STS benchmark dataset. The semantic similarity between the learning outcomes is determined by the cosine value between the embeddings as,

$$semantic\_similarity\ (lo1,\ lo2) = \frac{lo1 \cdot lo2}{\sqrt{\sum_{i=1}^{n} lo1}\sqrt{\sum_{i=1}^{n} lo2}} \tag{4.4}$$

The *semantic_similarity_grid* with the dimension of $m \times n$ is formed and the *semantic_similarity* values are added to the cells. The output from the two initial passes are fed to the final pass for aggregation.

Pass 3: Aggregation

The focus of the final pass is to provide flexibility in the aggregation process to enable the decision-making authorities to accommodate the variations in the administrative process across different institutions. Three important tunable parameters are provided to adjust the level of leniency offered by the decision-making authority in providing credits namely, '*impact*', '*sim_threshold*', and '*lo_threshold*'. Given the taxonomic similarity grid and the semantic similarity grid from Pass 1 and Pass 2 the *impact* parameter determines the percentage of contribution of both the similarities to the overall similarity. The '*sim_threshold*' defines the value above which two learning outcomes are considered to be similar, and finally, the '*lo_threshold*' determines the number of learning outcomes that need to be similar in order to consider the courses in comparison to being similar. The higher the value of these three parameters the lesser the leniency in the decision-making process.

The *final_similarity_grid* is built by aggregating the values from the previous modules, in the ratio determined by the '*impact*' parameter as shown in Figure 4.3. The LOs along the rows of the grid belong to the receiving institution's course hence traversing along the rows, the maximum value in the cells is checked against the '*sim_threshold*' value to determine if the LO in the row is similar to any LO in the columns. The course level similarity is derived by checking if the number of learning outcomes having a similar counterpart meets the '*lo_threshold*'.

4.4 Dataset and Results

Benchmark Dataset

One of the major challenges in the given field of research is the absence of benchmark datasets to compare the performance of the proposed system. Although there are existing pathways developed manually, previous research show that most of them are influenced by bias (based on the reputation of institutions, year of study, and so on) and administrative accommodations [153]. To create a benchmark dataset devoid of bias, a survey was conducted among domain experts to analyze and annotate the similarity between courses from two different institutions. A survey with learning outcomes of 7 pairs of courses (sending and receiving) from the computer science domain was distributed among instructors from the department of computer science at a comprehensive research university in Canada. In order to avoid bias, the names of both the institutions were anonymized and explicit instructions were provided to the annotators to assume a neutral position. Although the survey was circulated among 14 professors only 5 responses were received. This lack of responses is mainly attributed to the fact that most faculties are not involved in the transfer pathway development process and it is carried out widely as an administrative task. The survey questionnaire consisted of questions to mark the similarity between the courses over a scale of 1 to 10 and a binary response (*'yes'* and *'no'*) for whether or not credit should be offered to the receiving course. The course pairs were annotated with a final decision value based on the maximum number of *'yes'* or *'no'* responses received from the annotators. The results of the survey are tabulated in Table 4.1 where the responses *'yes'* and *'no'* are color coded in *'green'* and *'red'* respectively. One of the interesting inferences from the survey results is the agreement between the responses in the threshold value of similarity above which the annotators were willing to offer credit. 4 out of 5 annotators offered credit only if they considered the similarity between the courses falls above 7 on the given scale of 1 to 10. It is also interesting to note that in spite of having no information other than the learning outcomes the annotators differed in their level of leniency which inspired the need to offer flexibility to control leniency in the proposed methodology. For example, from Table 4.1 it is evident that while annotator *'A2'* has followed a more lenient approach and offered credit for 6 out of 7 courses, annotator *'A5'* has adopted a more strict approach by offering credit for only 1 out of the 7 courses.

Table 4.1: Survey results for the proposed benchmark dataset.

	A1	**A2**	**A3**	**A4**	**A5**	**Final Annotation**
1	**COMP4121**	**COMP4121**	**COMP4121**	**COMP4121**	**COMP4121**	**COMP4121**
2	**COMP5341**	**COMP5341**	**COMP5341**	**COMP5341**	**COMP5341**	**COMP5341**
3	**COMP3321**	**COMP3321**	**COMP3321**	**COMP3321**	**COMP3321**	**COMP3321**
4	**COMP4321**	**COMP4321**	**COMP4321**	**COMP4321**	**COMP4321**	**COMP4321**
5	**COMP3519**	**COMP3519**	**COMP3519**	**COMP3519**	**COMP3519**	**COMP3519**
6	**COMP5470**	**COMP5470**	**COMP5470**	**COMP5470**	**COMP5470**	**COMP5470**
7	**COMP5471**	**COMP5471**	**COMP5471**	**COMP5471**	**COMP5471**	**COMP5471**

Results

The results of the proposed model are provided in Table 4.2. In order to provide context to the need for the proposed methodology, the results of the model are compared to the results obtained when only the semantic similarity of the learning outcomes is considered. The proposed model at a neutral setting achieves 85.74% agreement with the human annotation by annotating 6 out of the 7 credit decision correctly, while the semantic similarity model achieves only 54.75% agreement. For the neutral setting of the proposed model, the three parameters in the aggregation pass are set as follows. The *impact* parameter is set at 0.30 meaning that the semantic similarity contributes 70% to the overall similarity and the taxonomic similarity contributes to the remaining 30%. The *sim_threshold* is set at 0.65, meaning that the overall similarity should be more than 65% in order for the learning outcomes to be similar to each other. The *lo_threshold* is set at 0.5 which considers that at least half of the available learning outcomes have similar counterparts. In order to demonstrate the options for flexibility, the proposed model is run by modifying the *lo_threshold* parameter. As shown in Table 4.3 for the lenient setting the *lo_threshold* parameter is set at 0.33 and 0.66 and the model achieves an agreement of 85% with the most lenient anno-

Table 4.2: Performance comparison of the proposed model by with human annotation.

S.No	**Human**	**Semantic Similarity**	**Proposed Methodology Neutral**
1	COMP4121	COMP4121	COMP4121
2	COMP5341	COMP5341	COMP5341
3	COMP3321	COMP3321	COMP3321
4	COMP4321	COMP4321	COMP4321
5	COMP3519	COMP3519	COMP3519
6	COMP5470	COMP5470	COMP5470
7	COMP5471	COMP5471	COMP5471
Agreement out of 7 comparisons		4	**6**

tator and the most strict annotator, respectively. Similarly, lowering the *impact* parameter makes the model more aligned to being strict and vice versa. However, it is important to understand that increasing the impact of taxonomic similarity to a higher percentage results in determining the overall similarity based on only two or three action words in the learning outcomes. Decreasing the *sim_threshold* will make the model more lenient and increasing it will make the model more strict. The results of the model at various parameter settings are provided in the Appendix.

4.5 Challenges and Future Works

One of the important limitations of this research is the fewer number of data points in the benchmark proposed which is attributed to the limited availability of quality learning outcomes, limited response from domain expert annotators, and cost. It is pertinent to understand that there are unique challenges to be addressed in the attempt to automate ar-

Table 4.3: Performance of proposed models on varying the *lo_threshold* parameter.

S.No	**Most lenient annotator**	**Proposed methodology Lenient**	**Most strict annotator**	**Proposed methodology Strict**
1	**COMP4121**	**COMP4121**	**COMP4121**	**COMP4121**
2	**COMP5341**	**COMP5341**	**COMP5341**	**COMP5341**
3	**COMP3321**	**COMP3321**	**COMP3321**	**COMP3321**
4	**COMP4321**	**COMP4321**	**COMP4321**	**COMP4321**
5	**COMP3519**	**COMP3519**	**COMP3519**	**COMP3519**
6	**COMP5470**	**COMP5470**	**COMP5470**	**COMP5470**
7	**COMP5471**	**COMP5471**	**COMP5471**	**COMP5471**
Agreement out of 7 comparisons		**6**		**6**

ticulations. Although the existing transformer-based models achieve near-perfect results in benchmark datasets, a thorough understanding of these datasets brings to light one of their major shortcomings. Rogers et al. [126] conclude in their survey with a clear statement on the limitations of the benchmark datasets, "As with any optimization method, if there is a shortcut in the data, we have no reason to expect BERT to not learn it. But harder datasets that cannot be resolved with shallow heuristics are unlikely to emerge if their development is not as valued as modeling work." Chandrasekaran et al. [28] established the significant drop in performance of these models with the increase in complexity of sentences. The comparison of the complexity of the learning outcomes in the proposed benchmark dataset to the sentences in the STS dataset is shown in Figure 4.5, which clearly indicates that learning outcomes are more complex. Also, the introduction of domain-specific BERT models shows clear indications that though the transformer models are trained using significantly large corpora with millions of words in their vocabularies, a domain-specific corpus is required to achieve better results in domain-specific tasks. Learning outcomes are not only complex

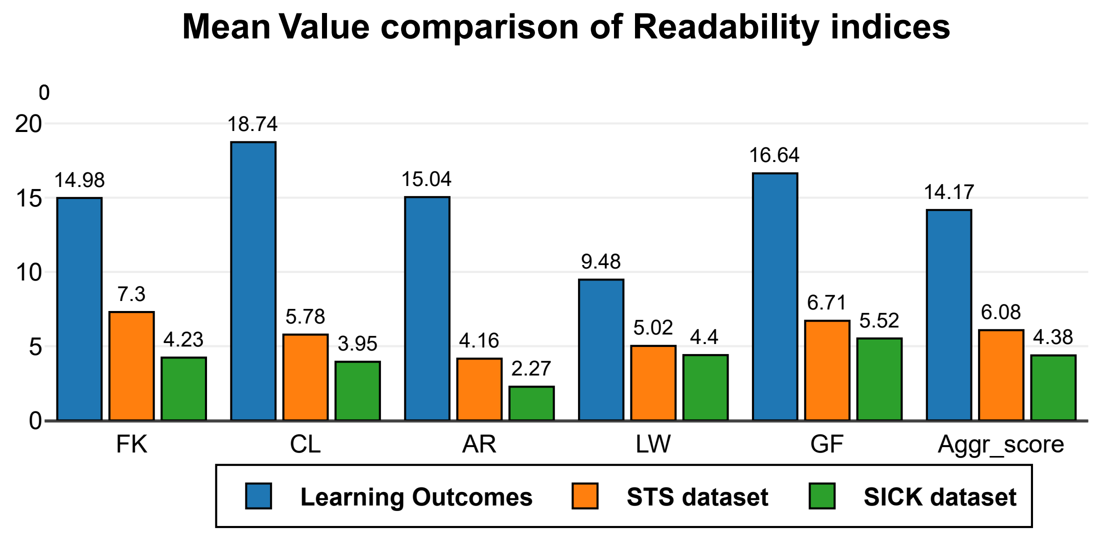


Figure 4.5: Comparison of the readability indices of the learning outcomes used in the proposed dataset and the existing benchmark datasets (STS and SICK dataset).

sentences but also contain domain-specific terminologies from various domains. Identifying these research gaps in semantic similarity methods is essential to contextualize and focus future research on addressing them. In addition to the technological challenges, it is also important to understand the challenges faced in the field of articulation. One of the approaches to enhance the performance of existing semantic similarity models is to train them with a large dataset of learning outcomes with annotated similarity values. However, learning outcomes are often considered to be intellectual properties of the instructors and are not publicly available. While almost all universities focus on building quality learning outcomes, most learning outcomes are either vague or don't follow the structural requirements of learning outcomes [136]. Even if a significant amount of learning outcomes are collected, annotation of their similarity requires expertise in subject matter and understanding of the articulation process. This annotation process is considerably more expensive than the annotation of English sentences as well. For example, one of the popular crowdsourcing platforms charges $0.04 for annotators with no specification and $0.65 for an annotator with at least a Master's degree. Furthermore, the selection of annotators with the required expertise needs manual scrutiny using preliminary questionnaires and surveys which makes the process time-consuming. Finally, articulation agreements are developed across different departments and it is imminent to provide a clear understanding of the model and its limitations to encourage automation of the process.The proposed model allows transparency and flexibility in the assessment of credit transfer and future research will focus on addressing these limitations by adding more course-to-course comparisons to the benchmark, developing domain-specific corpora, tuning the semantic similarity models with the aid of datasets and also on ways to improve the generation of learning outcomes through automation.

4.6 Conclusion

The assessment of transfer credit in the process of student mobility is considered to be one of the most crucial and time-consuming tasks, and across the globe, various steps have been taken to mitigate this process. With significant research and advancements in the field of natural language processing, this research article attempts to automate the articulation process by measuring the semantic and taxonomic similarity between learning outcomes. The proposed model uses a recent transformer-based language model to measure the semantic similarity and a clustering-inspired methodology is proposed to measure the taxonomic similarity of the learning outcomes. The model also comprises a flexible aggregation module to aggregate the similarity between learning outcomes to course-level learning outcomes. A benchmark dataset is built by conducting a survey among academicians to annotate the similarity and transfer credit decisions between courses from two different institutions. The results of the proposed model are compared with those of the benchmark dataset at different settings of leniency. The article also identifies the technical and domain-specific challenges that should be addressed in the field of automating articulation.

Acknowledgement

I would like to thank Dr. Thangarajah Akilan, Dr. Salimur Choudhury, Dr. Quazi Abidur Rahman, Dr. Yimin Yang, and Dr. Vijay Mago for helping this research by participating in the survey to build the benchmark dataset. Thanks for your valued time and expertise without which this research would not have possible.

Chapter 5

Conclusion

In this thesis, the evolution of semantic similarity over the past decades is traced by surveying over 100 peer-reviewed articles in the literature review chapter. The thesis begins with an in-depth analysis of the semantic similarity models classifying them as knowledge-based, corpus-based, deep neural network-based, and hybrid methods based on the underlying principle used to determine the semantic similarity values. The pros and cons of each method are described highlighting the need for each model in different scenarios. Knowledge-based methods take into consideration the actual meaning of text however, they are not adaptable across different domains and languages. Corpus-based methods have a statistical background and can be implemented across languages but they do not take into consideration the actual meaning of the text. Deep neural network-based methods show better performance, but they require high computational resources and lack interpretability. Hybrid methods are formed to take advantage of the benefits from different methods compensating for the shortcomings of each other. It is clear from the survey that each method has its advantages and disadvantages and it is difficult to choose one best model. The next chapter highlights one of the research gaps in the existing literature, by building a new benchmark dataset that proves the hypothesis that the performance of language models decreases with the increase in complexity of sentences. The proposed dataset consists of fifty complex sentence pairs and corresponding human-annotated semantic similarity values. The performance of various state-of-the-art models on the proposed dataset is compared. With a clear understanding of the advancements and limitations of the existing semantic similarity models, the final chapter explores one of the applications of the models in real-world tasks. In the process of student mobility, assessing transfer credit for incoming students is one of the important and laborious tasks. Hence, the proposed model attempts to use the high-performing transformer-based language models to calculate the semantic similarity and uses the knowledge-based semantic similarity methods in a clustering-inspired methodology to assess the taxonomic similarity of learning outcomes of courses from different institutions. Finally, these similarities are aggregated to form course-to-course similarity. The aggregation module of the model offers flexibility to the decision-making authorities to offer various levels of flexibility. In addition to using the existing methods to effectively automate articulation, the thesis also highlights the challenges in the domain that paves the direction of future research.

Appendix A

Algorithms

Algorithm 1 Best semantic similarity measure for verbs.

Input: $\mathcal{D}_{verb}[v_1, v_2, s]$ = SimVerb3500 dataset containing verb pairs (v_1, v_2) and associated semantic similarity benchmark values s,
wv = pretrained *word2vec* word embedding model
gl = pretrained $GloVe$ word embedding model
wn = $WordNet$ lexical knowledge base
Output: *best_measure* = the semantic similarity measure that achieves the best correlation to the benchmark values.

```
 1: $\mathcal{S} \leftarrow \varnothing$
 2: procedure VERB_SIMILARITY
 3:     for each $(v_1, v_2, s) \in \mathcal{D}_{verb}$ do
 4:         $\mathcal{S} += s$
 5:         $w1 \leftarrow wv.embedding(v_1)$
 6:         $w2 \leftarrow wv.embedding(v_2)$
 7:         $g1 \leftarrow gl.embedding(v_1)$
 8:         $g2 \leftarrow gl.embedding(v_2)$
 9:         $synsets1[syn_{11}...syn_{1i}] \leftarrow wn.get\_synsets(v_1, POS: verb)$    ▷ List of synsets
10:         $synsets2[syn_{21}....syn_{2j}] \leftarrow wn.get\_synsets(v_2, POS: verb)$    ▷ List of synsets
11:         for each $syn_{1i} \in synsets1$ do
12:             for each $syn_{2j} \in synsets2$ do
13:                 $wup\_sim\_list[syn_{1i}][syn_{2j}] \leftarrow cal\_wup(syn_1, syn_2)$    ▷ wup_similarity
14:                 $path\_sim\_list[syn_{1i}][syn_{2j}] \leftarrow cal\_path(syn_1, syn_2)$    ▷ path_similarity
15:                 $lch\_sim\_list[syn_{1i}][syn_{2j}] \leftarrow cal\_lch(syn_1, syn_2)$    ▷ lch_similarity
16:             end for
17:         end for
18:         $wup\_sim \leftarrow max(wup\_sim\_list[n])$
19:         $path\_sim \leftarrow max(path\_sim\_list[n])$
20:         $lch\_sim \leftarrow max(lch\_sim\_list[n])$
21:         $word2vec\_sim \leftarrow wv.similarity(w1, w2)$
22:         $glove\_sim \leftarrow gl.similarity(g1, g2)$
23:     end for
24:     $measure\_list += wup\_sim, path\_sim, lch\_sim, word2vec\_sim, glove\_sim$
25:     for each $m \in measure\_list$ do
26:         $cor \leftarrow get\_pearson\_correlation(S, m)$
27:     end for
28:     $best\_measure \leftarrow max(cor)$
29:     return $best\_measure$
30: end procedure
```

Algorithm 2 Pass_1: Taxonomic Similarity

Input: $\mathcal{S}lo\_list$ = List of m learning outcomes from the sending institution.
$\mathcal{R}lo\_list$ = List of n learning outcomes from the receiving institution.
$\mathcal{C}luster\_list = \{C_1, C_2, C_3, C_4, C_5, C_6\}$, List of six clusters initialized with illustrative verbs, such that $C_1 = \{v_1, v_2 ... v_n\}$ is a list of n verbs
Output: $taxonomic\_similarity\_grid$ = A $m \times n$ grid containing taxonomic similarity values between sending and receiving learning outcomes

```
1: procedure BUILD_TAXONOMIC_SIMILARITY_GRID
2:     for each $(slo) \in \mathcal{S}lo\_list$ do
3:         $slo\_verbs \leftarrow detect\_verbs(slo)$            ▷ Find the verbs in the learning outcomes
4:         for each $sv \in slo$ do
5:             if $sv \in \mathcal{C}luster\_list$ then
6:                 $\mathcal{B}(sv) \leftarrow get\_index(C_i)$        ▷ If the verb is among the predefined verbs
7:                                                     assign the corresponding cluster id.
8:             else
9:                 $\mathcal{B}(sv) \leftarrow get\_BestCluster(sv)$   ▷ Assign the cluster id of the best cluster
10:                                                    based on the silhouette width
11:            end if
12:        end for
13:        $\mathcal{B}(slo) \leftarrow max\big(\mathcal{B}(sv_1), ....\mathcal{B}(sv_n)\big)$
14:    end for
15:    for each $(rlo) \in \mathcal{R}lo\_list$ do                 ▷ Repeat for receiving LOs
16:        $rlo\_verbs \leftarrow detect\_verbs(rlo)$
17:        for each $rv \in rlo$ do
18:            if $rv \in \mathcal{C}luster\_list$ then
19:                $\mathcal{B}(rv) \leftarrow get\_index(C_i)$
20:            else
21:                $\mathcal{B}(rv) \leftarrow get\_BestCluster(rv)$
22:            end if
23:        end for
24:        $\mathcal{B}(rlo) \leftarrow max(\mathcal{B}(rv_1), ....\mathcal{B}(rv_n)$
25:    end for
26:    for each $(slo) \in \mathcal{S}lo\_list$ do                 ▷ Calculate taxonomic similarity
27:        for each $(rlo) \in \mathcal{R}lo\_list$ do
28:            $taxonomic\_similarity\_grid[slo][rlo] \leftarrow abs\big(\mathcal{B}(rlo) - \mathcal{B}(slo)\big)$
29:        end for
30:    end for
31:    return $taxonomic\_similarity\_grid$
32: end procedure
```

Algorithm 3 Pass_2: Semantic Similarity

Input: $\mathcal{S}lo\_list$ = List of m learning outcomes from the sending institution.
$\mathcal{R}lo\_list$ = List of n learning outcomes from the receiving institution.
$model$ = Pretrained and Fine tuned SRoBERTa transformer-based semantic similarity model **Output:** $semantic\_similarity\_grid$ = A $m \times n$ grid containing semantic similarity values between sending and receiving learning outcomes

```
1: procedure BUILD_SEMANTIC_SIMILARITY_GRID
2:     for each $slo, rlo \in \mathcal{S}lo\_list\mathcal{R}lo\_list$ do
3:         $\vec{slo} \leftarrow model.sentence\_vectors(slo)$        ▷ Get vector representation of
4:                                                                    learning outcomes
5:         $\vec{rlo} \leftarrow model.sentence\_vectors(rlo)$
6:                                                                  ▷ Calculate semantic similarity
7:         $semantic\_similarity\_grid[slo][rlo] \leftarrow cos(\vec{slo}, \vec{rlo})$
8:     end for
9:     return $semantic\_similarity\_grid$
10: end procedure
```

Algorithm 4 Pass_3: Aggregation

Input: $SSG = \textit{semantic\_similarity\_grid}$, a $m \times n$ grid containing semantic similarity values between sending and receiving learning outcomes
$TSG = \textit{taxonomic\_similarity\_grid}$, a $m \times n$ grid containing taxonomic similarity values between sending and receiving learning outcomes
α = parameter which determines the ratio of two similarity value in the overall similarity
β = value above which two learning outcomes are considered similar
γ = number of learning outcomes with similar counterparts
Output: $\mathcal{TC}$ = Final Credit Decision

```
1: procedure COURSE_LEVEL_SIMILARITY
2:     for each $row, col \in SSG$ do
3:         for each $row, col \in TSG$ do
4:             $final\_sim \leftarrow \big(\alpha \times SSG[row, col] + (1 - \alpha) \times TSG[row, col]\big)$
5:             if $final\_sim >= \beta$ then
6:                 $final\_similarity\_grid[row, column] \leftarrow TRUE$
7:             else
8:                 $final\_similarity\_grid[row, column] \leftarrow FALSE$
9:             end if
10:        end for
11:    end for
12:    for each $row \in final\_similarity\_grid$ do
13:        if $Count(TRUE) >= \gamma$ then
14:            $\mathcal{TC} = Yes$
15:        else
16:            $\mathcal{TC} = No$
17:        end if
18:    end for
19:    return $\mathcal{TC}$
20: end procedure
```

Appendix B

Semantic Similarity Benchmark Datasets

This section discusses some of the popular datasets used to evaluate the performance of semantic similarity algorithms. The datasets may include word pairs or sentence pairs with associated standard similarity values. The performance of various semantic similarity algorithms is measured by the correlation of the achieved results with that of the standard measures available in these datasets. Table B.1 lists some of the popular datasets used to evaluate the performance of semantic similarity algorithms. The below subsection describes the attributes of the dataset and the methodology used to construct them.

Dataset Name	**Word/Sentence pairs**	**Similarity score range**	**Year**	**Reference**
R&G	65	0-4	1965	[129]
M&C	30	0-4	1991	[92]
WS353	353	0-10	2002	[36]
LiSent	65	0-4	2007	[75]

Table B.1 continued from previous page

Dataset Name	**Word/Sentence pairs**	**Similarity score range**	**Year**	**Reference**
SRS	30	0-4	2007	[109]
WS353-Sim	203	0-10	2009	[2]
STS2012	5250	0-5	2012	[3]
STS2013	2250	0-5	2013	[1]
WP300	300	0-1	2013	[73]
STS2014	3750	0-5	2014	[4]
SL7576	7576	1-5	2014	[143]
SimLex-999	999	0-10	2014	[48]
SICK	10000	1-5	2014	[82]
STS2015	3000	0-5	2015	[5]
SimVerb	3500	0-10	2016	[40]
STS2016	1186	0-5	2016	[6]
WiC	5428	NA	2019	[112]

Table B.1: Popular benchmark datasets for Semantic similarity.

Semantic similarity datasets

The following is a list of widely used semantic similarity datasets arranged chronologically.

- **Rubenstein and Goodenough (R&G)** [129]: This dataset was created as a result of an experiment conducted among 51 undergraduate students (native English speakers) in two different sessions. The subjects were provided with 65 selected English noun pairs and requested to assign a similarity score for each pair over a scale of 0 to 4, where 0 represents that the words are completely dissimilar and 4 represents that they are highly similar. This dataset is the first and most widely used dataset in semantic similarity tasks [164].

- **Miller and Charles (M&C)** [92]: Miller and Charles repeated the experiment performed by Rubenstein and Goodenough in 1991 with a subset of 30 word pairs from the original 65 word pairs. 38 human subjects ranked the word pairs on a scale from 0 to 4, 4 being the "most similar."

- **WS353** [36]: WS353 contains 353 word pairs with an associated score ranging from 0 to 10. 0 represents the least similarity and 10 represents the highest similarity. The experiment was conducted with a group of 16 human subjects. This dataset measures semantic relatedness rather than semantic similarity. Subsequently, the next dataset was proposed.

- **WS353-Sim** [2]: This dataset is a subset of WS353 containing 203 word pairs from the original 353 word pairs that are more suitable for semantic similarity algorithms specifically.

- **LiSent** [75]: 65 sentence pairs were built using the dictionary definition of 65 word pairs used in the R&G dataset. 32 native English speakers volunteered to provide a similarity range from 0 to 4, 4 being the highest. The mean of the scores given by all the volunteers was taken as the final score.

- **SRS** [109]: Pedersen et al. [109] attempted to build a domain specific semantic similarity dataset for the biomedical domain. Initially 120 pairs were selected by a physician distributed with 30 pairs over 4 similarity values. These term pairs were then ranked by 13 medical coders on a scale of 1-10. 30 word pairs from the 120 pairs were selected to increase reliability and these word pairs were annotated by 3 physicians and 9 (out of the 13) medical coders to form the final dataset.

- **SimLex-999** [48]: 999 word pairs were selected from the UFS Dataset [103] of which 900 were similar and 99 were related but not similar. 500 native English speakers, recruited via Amazon Mechanical Turk were asked to rank the similarity between the word pairs over a scale of 0 to 6, 6 being the most similar. The dataset contains 666 noun pairs, 222 verb pairs, and 111 adjective pairs.

- **Sentences Involving Compositional Knowledge (SICK) dataset** [82]: The SICK dataset consists of 10,000 sentence pairs, derived from two existing datasets the ImageFlickr 8 and MSR-Video descriptions dataset. Each sentence pair is associated with a relatedness score and a text entailment relation. The relatedness score ranges from 1 to 5, and the three entailment relations are "NEUTRAL, ENTAILMENT and CONTRADICTION." The annotation was done using crowd-sourcing techniques.

- **STS datasets** [3, 1, 4, 5, 6, 27]: The STS datasets were built by combining sentence pairs from different sources by the organizers of the SemEVAL shared task. The dataset was annotated using Amazon Mechanical Turk and further verified by the organizers themselves. Table B.2 shows the various sources from which the STS dataset was built.

Year	**Dataset**	**Pairs**	**Source**
2012	MSRPar	1500	newswire
2012	MSRvid	1500	videos
2012	OnWN	750	glosses
2012	SMTNews	750	WMT eval.
2012	SMTeuroparl	750	WMT eval.
2013	HDL	750	newswire
2013	FNWN	189	glosses
2013	OnWN	561	glosses
2013	SMT	750	MT eval.
2014	HDL	750	newswire headlines
2014	OnWN	750	glosses
2014	Deft-forum	450	forum posts
2014	Deft-news	300	news summary
2014	Images	750	image descriptions
2014	Tweet-news	750	tweet-news pairs
2015	HDL	750	newswire headlines
2015	Images	750	image descriptions
2015	Ans.-student	750	student answers
2015	Ans.-forum	375	Q & A forum answers
2015	Belief	375	committed belief
2016	HDL	249	newswire headlines
2016	Plagiarism	230	short-answers plag.
2016	post-editing	244	MT postedits

Table B.2 continued from previous page

Year	**Dataset**	**Pairs**	**Source**
2016	Ans.-Ans	254	Q & A forum answers
2016	Quest.-Quest.	209	Q & A forum questions
2017	Trail	23	Mixed STS 2016

Table B.2: STS English language training dataset (2012-2017) [27].

Appendix C

Semantic distance measures and their formulae

SNo	**Semantic distance measure**	**Formula**
1	α - skew divergence (ASD)	$\sum_{w \in C(w_1) \cup C(w_2)} P(w\|w_1) log \frac{P(w\|w_1)}{\alpha P(w\|w_2) + (1-\alpha) P(w\|w_1)}$
2	Cosine similarity	$\frac{\sum_{w \in C(w_1) \cup C(w_2)} P(w \mid w_1) \times P(w \mid w_2)}{\sqrt{\sum_{w \in C(w_1)} P(w \mid w_1)^2} \times \sqrt{\sum_{w \in C(w_2)} P(w\|w_2)^2}}$
3	Co-occurence Retrieval Models (CRM)	$\gamma \left[\frac{2 \times P \times R}{P + R} \right] + (1-\gamma) \left[\beta[P] + (1-\beta)[R] \right]$

Table C.1 continued from previous page

SNo	**Semantic distance measure**	**Formula**
4	Dice coefficient	$\frac{2 \times \sum_{w \in C(w_1) \cup C(w_2)} \min(P(w\|w_1), P(w\|w_2))}{\sum_{w \in C(w_1)} P(w\|w_1) + \sum_{w \in C(w_2)} P(w\|w_2)}$
5	Manhattan Distance or L1 norm	$\sum_{w \in C(w_1) \cup C(w_2)} \left\| P(w\|w_1) - P(w\|w_2) \right\|$
6	Division measure	$\sum_{w \in C(w_1) \cup C(w_2)} \left\| \log \frac{P(w\|w_1)}{P(w\|w_2)} \right\|$
7	Hindle	$\sum_{w \in C(w)} \begin{cases} \min(I(w,w_1), I(w,w_2)), \\ \quad \text{if both} I(w,w_1) \text{and} I(w,w_2) > 0 \\ \|\max(I(w,w_1), I(w,w_2))\|, \\ \quad \text{if both} I(w,w_1) \text{and} I(w,w_2) < 0 \\ 0, \text{otherwise} \end{cases}$
8	Jaccard	$\frac{\sum_{w \in C(w_1) \cup C(w_2)} \min(P(w\|w_1), P(w\|w_2))}{\sum_{w \in C(w_1) \cup C(w_2)} \max(P(w\|w_1), P(w\|w_2))}$

Table C.1 continued from previous page

SNo	**Semantic distance measure**	**Formula**
9	Jensen-Shannon divergence (JSD)	$\sum_{w \in C(w_1) \cup C(w_2)} \left(P(w\|w_1) \log \frac{P(w\|w_1)}{\frac{1}{2}(P(w\|w_1)+P(w\|w_2))} + P(w\|w_2) \log \frac{P(w\|w_2)}{\frac{1}{2}(P(w\|w_1)+P(w\|w_2))} \right)$
10	Kullback-Leibler divergence - common occurance	$\sum_{w \in C(w_1) \cup C(w_2)} P(w\|w_1) \log \frac{P(w\|w_1)}{P(w\|w_2)}$
11	Kullback-Leibler divergence - absolute	$\sum_{w \in C(w_1) \cup C(w_2)} P(w\|w_1) \left\| \log \frac{P(w\|w_1)}{P(w\|w_2)} \right\|$
12	Kullback-Leibler divergence - average	$\frac{1}{2} \sum_{w \in C(w_1) \cup C(w_2)} (P(w\|w_1) - P(w\|w_2)) \log \frac{P(w\|w_1)}{P(w\|w_2)}$

Table C.1 continued from previous page

SNo	**Semantic distance measure**	**Formula**
13	Kullback-Leibler divergence - maximum	$\max(KLD(w_1, w_2), KLD(w_2, w_1))$
14	Euclidean Distance or L2 norm	$\sqrt{\sum_{w \in C(w_1) \cup C(w_2)} (P(w\|w_1) - P(w\|w_2))^2}$
15	Lin	$\frac{\sum_{(r,w) \in T(w_1) \cap T(w_2)} (I(w_1, r, w) + I(w_2, r, w))}{\sum(r, w') \in T(w_1) I(w_1, r, w') + \sum_{(r,w'') \in T(w_2)} I(w_2, r, w'')}$
16	Product measure	$\sum_{w \in C(w_1) \cup C(w_2)} \frac{P(w\|w_1) \times P(w\|w_2)}{(\frac{1}{2}(P(w\|w_1) + P(w\|w_2)))^2}$

Table C.1: Table of semantic measures and their formulae - adapted from Mohammad and Hurst[95].

Table C.2: Performance of the proposed model on varying *sim_threshold* parameter.

S.No.	**Human Annotation**	**sim_threshold = 60**	**sim_threshold = 65**	**sim_threshold = 70**
1	**COMP4121**	**COMP4121**	**COMP4121**	**COMP4121**
2	**COMP5341**	**COMP5341**	**COMP5341**	**COMP5341**
3	**COMP3321**	**COMP3321**	**COMP3321**	**COMP3321**
4	**COMP4321**	**COMP4321**	**COMP4321**	**COMP4321**
5	**COMP3519**	**COMP3519**	**COMP3519**	**COMP3519**
6	**COMP5470**	**COMP5470**	**COMP5470**	**COMP5470**
7	**COMP5471**	**COMP5471**	**COMP5471**	**COMP5471**
Agreement out of 7 comparisons		4	**6**	4

Table C.3: Performance of the proposed model on varying *impact* parameter.

S.No.	**Human Annotation**	**impact = 20**	**impact = 30**	**impact = 40**
1	**COMP4121**	**COMP4121**	**COMP4121**	**COMP4121**
2	**COMP5341**	**COMP5341**	**COMP5341**	**COMP5341**
3	**COMP3321**	**COMP3321**	**COMP3321**	**COMP3321**
4	**COMP4321**	**COMP4321**	**COMP4321**	**COMP4321**
5	**COMP3519**	**COMP3519**	**COMP3519**	**COMP3519**
6	**COMP5470**	**COMP5470**	**COMP5470**	**COMP5470**
7	**COMP5471**	**COMP5471**	**COMP5471**	**COMP5471**
Agreement out of 7 comparisons		4	**6**	4

Appendix D

Abbreviations

Abbreviation	**Description**
NLP	Natural Language Processing
BoW	Bag of Words
AI	Artificial Intelligence
STS	Semantic Textual Similarity
LCS	Least Common Subsumer
IC	Information Content
TF-IDF	Term Frequency - Inverse Document Frequency
BERT	Bidirectional Encoder Representations from Transformers
AlBERT	A lite Bidirectional Encoder Representations from Transformers

RoBERTa	A Robustly Optimized BERT Pretraining approach
SICK	Sentences Involving Compositional Knowledge
NLI	Natural Language Inference
LO	Learning Outcome
PCA	Principal Component Analysis
GloVe	Global Vectors for Word Representation
CBoW	Continuous Bag of Words
LSTM	Long-Short Term Memory
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network

Bibliography

[1] Eneko Agirre et al. "* SEM 2013 shared task: Semantic textual similarity". In: *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*. 2013, pp. 32–43.

[2] Eneko Agirre et al. "A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches". In: *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2009, pp. 19–27.

[3] Eneko Agirre et al. "Semeval-2012 task 6: A pilot on semantic textual similarity". In: ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. 2012, pp. 385–393.

[4] Eneko Agirre et al. "Semeval-2014 task 10: Multilingual semantic textual similarity". In: *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*. 2014, pp. 81–91.

[5] Eneko Agirre et al. "Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability". In: *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015).* 2015, pp. 252–263.

[6] Eneko Agirre et al. "Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation". In: *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics). 2016.

[7] Jamal Al Qundus et al. "Exploring the impact of short-text complexity and structure on its quality in social media". In: *Journal of Enterprise Information Management* (2020).

[8] Alexander M. Rush, Sumit Chopra, and Jason Weston. "A Neural Attention Model for Abstractive Sentence". In: *Proceedings of the 2015 conference on empirical methods in natural language processing.* 5.3 (2015), pp. 379–389. ISSN: 2302-4496.

[9] Berna Altınel and Murat Can Ganiz. "Semantic text classification: A survey of past and recent advances". In: *Information Processing & Management* 54.6 (2018), pp. 1129–1153. ISSN: 0306-4573. DOI: `https://doi.org/10.1016/j.ipm.2018.08.001`. URL: `http://www.sciencedirect.com/science/article/pii/S0306457317305757`.

[10] Samir Amir, Adrian Tanasescu, and Djamel A Zighed. "Sentence similarity based on semantic kernels for intelligent text retrieval". In: *Journal of Intelligent Information Systems* 48.3 (2017), pp. 675–689.

[11] Lorin W Anderson, Benjamin Samuel Bloom, et al. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives.* Longman, 2001.

[12] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". English (US). In: 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015. Jan. 2015.

[13] Satanjeev Banerjee and Ted Pedersen. "Extended gloss overlaps as a measure of semantic relatedness". In: *Ijcai*. Vol. 3. 2003, pp. 805–810.

[14] Daniel Bär et al. "Ukp: Computing semantic textual similarity by combining multiple content similarity measures". In: ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. 2012, pp. 435–440.

[15] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2014, pp. 238–247.

[16] Marco Baroni et al. "The WaCky wide web: a collection of very large linguistically processed web-crawled corpora". In: *Language resources and evaluation* 43.3 (2009), pp. 209–226.

[17] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2019, pp. 3606–3611.

[18] Fabio Benedetti et al. "Computing inter-document similarity with Context Semantic Analysis". In: *Information Systems* 80 (2019), pp. 136–147. ISSN: 0306-4379.

DOI: https://doi.org/10.1016/j.is.2018.02.009. URL: http://www.sciencedirect.com/science/article/pii/S0306437917301503.

[19] Christian Bizer et al. “DBpedia-A crystallization point for the Web of Data”. In: *Journal of web semantics* 7.3 (2009), pp. 154–165.

[20] Benjamin S Bloom and David R Krathwohl. “Taxonomy of Educational Objectives: The Classification of Educational Goals”. In: *Handbook I: Cognitive Domain.* 1956.

[21] Piotr Bojanowski et al. “Enriching word vectors with subword information”. In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

[22] Antoine Bordes, Sumit Chopra, and Jason Weston. “Question Answering with Subgraph Embeddings”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* 2014, pp. 615–620.

[23] Jose Camacho-Collados and Mohammad Taher Pilehvar. “From word to sense embeddings: A survey on vector representations of meaning”. In: *Journal of Artificial Intelligence Research* 63 (2018), pp. 743–788.

[24] José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. “Nasari: a novel approach to a semantically-aware representation of items”. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 2015, pp. 567–577.

[25] José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. “Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities”. In: *Artificial Intelligence* 240 (2016), pp. 36–64. ISSN: 0004-

3702. DOI: `https://doi.org/10.1016/j.artint.2016.07.005`. URL: `http://www.sciencedirect.com/science/article/pii/S0004370216300820`.

[26] Nicola Cancedda et al. "Word-sequence kernels". In: *Journal of machine learning research* 3.Feb (2003), pp. 1059–1082.

[27] Daniel Cer et al. "SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 2017, pp. 1–14.

[28] Dhivya Chandrasekaran and Vijay Mago. "Domain Specific Complex Sentence (DCSC) Semantic Similarity Dataset". In: *arXiv preprint arXiv:2010.12637* (2020).

[29] Dhivya Chandrasekaran and Vijay Mago. "Evolution of Semantic Similarity-A Survey". In: *ACM Computing Surveys (CSUR)* 54.2 (2021), pp. 1–37.

[30] Rudi L Cilibrasi and Paul MB Vitanyi. "The google similarity distance". In: *IEEE Transactions on knowledge and data engineering* 19.3 (2007), pp. 370–383.

[31] Meri Coleman and Ta Lin Liau. "A computer readability formula designed for machine scoring." In: *Journal of Applied Psychology* 60.2 (1975), p. 283.

[32] Michael Collins and Nigel Duffy. "Convolution kernels for natural language". In: *Advances in neural information processing systems*. 2002, pp. 625–632.

[33] Michael Collins and Nigel Duffy. "New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 2002, pp. 263–270.

[34] Danilo Croce et al. "Deep learning in semantic kernel spaces". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2017, pp. 345–354.

[35] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4171–4186.

[36] Lev Finkelstein et al. "Placing search in context: The concept revisited". In: *Proceedings of the 10th international conference on World Wide Web*. 2001, pp. 406–414.

[37] Evgeniy Gabrilovich, Shaul Markovitch, et al. "Computing semantic relatedness using wikipedia-based explicit semantic analysis." In: *IJcAI*. Vol. 7. 2007, pp. 1606–1611.

[38] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. "PPDB: The paraphrase database". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2013, pp. 758–764.

[39] Jian-Bo Gao, Bao-Wen Zhang, and Xiao-Hua Chen. "A WordNet-based semantic similarity measurement combining edge-counting and information content theory". In: *Engineering Applications of Artificial Intelligence* 39 (2015), pp. 80–88. ISSN: 0952-1976. DOI: `https://doi.org/10.1016/j.engappai.2014.11.009`. URL: `http://www.sciencedirect.com/science/article/pii/S0952197614002814`.

[40] Daniela Gerz et al. "SimVerb-3500: A Large-Scale Evaluation Set of Verb Similarity". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2016, pp. 2173–2182.

[41] Goran Glavaš et al. "A resource-light method for cross-lingual semantic textual similarity". In: *Knowledge-Based Systems* 143 (2018), pp. 1–9. ISSN: 0950-7051. DOI:

https://doi.org/10.1016/j.knosys.2017.11.041. URL: http://www.sciencedirect.com/science/article/pii/S0950705117305725.

[42] James Gorman and James R Curran. "Scaling distributional similarity to large corpora". In: *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. 2006, pp. 361–368.

[43] Robert Gunning et al. "Technique of clear writing". In: (1952).

[44] Mohamed Ali Hadj Taieb, Torsten Zesch, and Mohamed Ben Aouicha. "A survey of semantic relatedness evaluation datasets and procedures". In: *Artificial Intelligence Review* (Dec. 2019). ISSN: 1573-7462. DOI: 10.1007/s10462-019-09796-3. URL: https://doi.org/10.1007/s10462-019-09796-3.

[45] Basma Hassan et al. "UESTS: An Unsupervised Ensemble Semantic Textual Similarity Method". In: *IEEE Access* 7 (2019), pp. 85462–85482.

[46] Hua He and Jimmy Lin. "Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 937–948. DOI: 10.18653/v1/N16-1108. URL: https://www.aclweb.org/anthology/N16-1108.

[47] A. Heppner et al. "Automating Articulation: Applying Natural Language Processing to Post-Secondary Credit Transfer". In: *IEEE Access* 7 (2019), pp. 48295–48306. DOI: 10.1109/ACCESS.2019.2910145.

[48] Felix Hill, Roi Reichart, and Anna Korhonen. "Simlex-999: Evaluating semantic models with (genuine) similarity estimation". In: *Computational Linguistics* 41.4 (2015), pp. 665–695.

[49] Johannes Hoffart et al. "YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia". In: *Artificial Intelligence* 194 (2013), pp. 28–61.

[50] Harneet Kaur Janda et al. "Syntactic, Semantic and Sentiment Analysis: The Joint Effect on Automated Essay Evaluation". In: *IEEE Access* 7 (2019), pp. 108486–108503.

[51] Jay J Jiang and David W Conrath. "Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy". In: *Proceedings of the 10th Research on Computational Linguistics International Conference*. 1997, pp. 19–33.

[52] Yuncheng Jiang et al. "Feature-based approaches to semantic similarity assessment of concepts using Wikipedia". In: *Information Processing & Management* 51.3 (2015), pp. 215–234.

[53] Yuncheng Jiang et al. "Wikipedia-based information content and semantic similarity computation". In: *Information Processing & Management* 53.1 (2017), pp. 248–265. ISSN: 0306-4573. DOI: `https://doi.org/10.1016/j.ipm.2016.09.001`. URL: `http://www.sciencedirect.com/science/article/pii/S0306457316303934`.

[54] Xiaoqi Jiao et al. "Tinybert: Distilling bert for natural language understanding". In: *arXiv preprint arXiv:1909.10351* (2019).

[55] "Junor, S., & Usher, A. (2008). Student Mobility & Credit Transfer: A National and Global Survey." In: 2008.

[56] Tomoyuki Kajiwara and Mamoru Komachi. "Building a monolingual parallel corpus for text simplification using sentence similarity based on alignment between word

embeddings”. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016, pp. 1147–1158.

[57] Sun Kim et al. “Bridging the gap: Incorporating a semantic similarity measure for effectively mapping PubMed queries to documents”. In: *Journal of biomedical informatics* 75 (2017), pp. 122–127.

[58] Yoon Kim. “Convolutional Neural Networks for Sentence Classification”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1746–1751.

[59] J Peter Kincaid et al. *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Tech. rep. Naval Technical Training Command Millington TN Research Branch, 1975.

[60] Zhenzhong Lan et al. “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations”. In: *International Conference on Learning Representations*. 2019.

[61] Thomas K Landauer and Susan T Dumais. “A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.” In: *Psychological review* 104.2 (1997), p. 211.

[62] Thomas K Landauer, Peter W Foltz, and Darrell Laham. “An introduction to latent semantic analysis”. In: *Discourse processes* 25.2-3 (1998), pp. 259–284.

[63] Juan J. Lastra-Díaz et al. “A reproducible survey on word embeddings and ontology-based methods for word similarity: Linear combinations outperform the state of the art”. In: *Engineering Applications of Artificial Intelligence* 85 (2019), pp. 645–665.

ISSN: 0952-1976. DOI: `https://doi.org/10.1016/j.engappai.2019.07.010`. URL: `http://www.sciencedirect.com/science/article/pii/S0952197619301745`.

[64] Juan J Lastra-Dıéaz and Ana Garcıéa-Serrano. "A new family of information content models with an experimental survey on WordNet". In: *Knowledge-Based Systems* 89 (2015), pp. 509–526.

[65] Juan J Lastra-Dıéaz et al. "HESML: A scalable ontology-based semantic similarity measures library with a set of reproducible experiments and a replication dataset". In: *Information Systems* 66 (2017), pp. 97–118.

[66] Quoc Le and Tomas Mikolov. "Distributed representations of sentences and documents". In: *International conference on machine learning*. 2014, pp. 1188–1196.

[67] Yuquan Le et al. "ACV-tree: A New Method for Sentence Similarity Modeling." In: *IJCAI*. 2018, pp. 4137–4143.

[68] Claudia Leacock and Martin Chodorow. "Combining local context and WordNet similarity for word sense identification". In: *WordNet: An electronic lexical database* 49.2 (1998), pp. 265–283.

[69] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4 (2020), pp. 1234–1240.

[70] Ming Che Lee. "A novel sentence similarity measure for semantic-based expert systems". In: *Expert Systems with Applications* 38.5 (2011), pp. 6392–6399.

[71] Omer Levy and Yoav Goldberg. "Dependency-based word embeddings". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 2014, pp. 302–308.

[72] Omer Levy and Yoav Goldberg. "Neural word embedding as implicit matrix factorization". In: *Advances in neural information processing systems*. 2014, pp. 2177–2185.

[73] Peipei Li et al. "Computing term similarity by large probabilistic isa knowledge". In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2013, pp. 1401–1410.

[74] Yuhua Li, Zuhair A Bandar, and David McLean. "An approach for measuring semantic similarity between words using multiple information sources". In: *IEEE Transactions on knowledge and data engineering* 15.4 (2003), pp. 871–882.

[75] Yuhua Li et al. "Sentence similarity based on semantic nets and corpus statistics". In: *IEEE transactions on knowledge and data engineering* 18.8 (2006), pp. 1138–1150.

[76] Dekang Lin et al. "An information-theoretic definition of similarity." In: *Icml*. Vol. 98. 1998. 1998, pp. 296–304.

[77] Yang Liu et al. "Topical word embeddings". In: *Twenty-ninth AAAI conference on artificial intelligence*. Citeseer. 2015.

[78] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[79] I. Lopez-Gazpio et al. "Interpretable semantic textual similarity: Finding and explaining differences between sentences". In: *Knowledge-Based Systems* 119 (2017), pp. 186–199. ISSN: 0950-7051. DOI: `https://doi.org/10.1016/j.knosys.2016.12.013`. URL: `http://www.sciencedirect.com/science/article/pii/S095070511630507X`.

[80] I. Lopez-Gazpio et al. "Word n-gram attention models for sentence similarity and inference". In: *Expert Systems with Applications* 132 (2019), pp. 1–11. ISSN: 0957-

4174. DOI: https://doi.org/10.1016/j.eswa.2019.04.054. URL: http://www.sciencedirect.com/science/article/pii/S0957417419302842.

[81] Kevin Lund and Curt Burgess. "Producing high-dimensional semantic spaces from lexical co-occurrence". In: *Behavior research methods, instruments, & computers* 28.2 (1996), pp. 203–208.

[82] Marco Marelli et al. "A SICK cure for the evaluation of compositional distributional semantic models". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), May 2014, pp. 216–223. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/363_Paper.pdf.

[83] Marco Marelli et al. "A SICK cure for the evaluation of compositional distributional semantic models." In: *LREC*. 2014, pp. 216–223.

[84] Bryan McCann et al. "Learned in translation: contextualized word vectors". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Curran Associates Inc. 2017, pp. 6297–6308.

[85] Bridget T McInnes et al. "UMLS:: Similarity: Measuring the Relatedness and Similarity of Biomedical Concepts". In: *Human Language Technologies: The 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, p. 28.

[86] Christopher Meek, Yang Yi, and Yih Wen-tau. "WIKIQA : A Challenge Dataset for Open-Domain Question Answering". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* September 2015 (2018), pp. 2013–2018. DOI: 10.18653/v1/D15-1237. URL: https://www.aclweb.org/anthology/D15-1237.

[87] Oren Melamud, Jacob Goldberger, and Ido Dagan. "context2vec: Learning generic context embedding with bidirectional LSTM". In: *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 2016, pp. 51–61.

[88] Rada Mihalcea and Andras Csomai. "Wikify! Linking documents to encyclopedic knowledge". In: *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. 2007, pp. 233–242.

[89] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[90] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. "Linguistic regularities in continuous space word representations". In: *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 2013, pp. 746–751.

[91] George A Miller. "WordNet: a lexical database for English". In: *Communications of the ACM* 38.11 (1995), pp. 39–41.

[92] George A Miller and Walter G Charles. "Contextual correlates of semantic similarity". In: *Language and cognitive processes* 6.1 (1991), pp. 1–28.

[93] Andriy Mnih and Koray Kavukcuoglu. "Learning word embeddings efficiently with noise-contrastive estimation". In: *Advances in neural information processing systems*. 2013, pp. 2265–2273.

[94] Muhidin Mohamed and Mourad Oussalah. "SRL-ESA-TextSum: A text summarization approach based on semantic role labeling and explicit semantic analysis". In: *Information Processing & Management* 56.4 (2019), pp. 1356–1372.

[95] Saif M Mohammad and Graeme Hirst. "Distributional measures of semantic distance: A survey". In: *arXiv preprint arXiv:1203.1858* (2012).

[96] Alessandro Moschitti. "Efficient convolution kernels for dependency and constituent syntactic trees". In: *European Conference on Machine Learning.* Springer. 2006, pp. 318–329.

[97] Alessandro Moschitti. "Kernel methods, syntax and semantics for relational text categorization". In: *Proceedings of the 17th ACM conference on Information and knowledge management.* 2008, pp. 253–262.

[98] Alessandro Moschitti, Daniele Pighin, and Roberto Basili. "Tree kernels for semantic role labeling". In: *Computational Linguistics* 34.2 (2008), pp. 193–224.

[99] Alessandro Moschitti and Silvia Quarteroni. "Kernels on linguistic structures for answer extraction". In: *Proceedings of ACL-08: HLT, Short Papers.* 2008, pp. 113–116.

[100] Alessandro Moschitti and Fabio Massimo Zanzotto. "Fast and effective kernels for relational learning from texts". In: *Proceedings of the 24th international conference on Machine learning.* 2007, pp. 649–656.

[101] Alessandro Moschitti et al. "Exploiting syntactic and shallow semantic kernels for question answer classification". In: *Proceedings of the 45th annual meeting of the association of computational linguistics.* 2007, pp. 776–783.

[102] Roberto Navigli and Simone Paolo Ponzetto. "BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network". In: *Artificial Intelligence* 193 (2012).

[103] Douglas L Nelson, Cathy L McEvoy, and Thomas A Schreiber. "The University of South Florida free association, rhyme, and word fragment norms". In: *Behavior Research Methods, Instruments, & Computers* 36.3 (2004), pp. 402–407.

[104] Joakim Nivre. *Inductive dependency parsing*. Springer, 2006.

[105] OECD. *What is the profile of internationally mobile students?* 2019. DOI: `https://doi.org/https://doi.org/10.1787/17d19cd9-en`. URL: `https://www.oecd-ilibrary.org/content/component/17d19cd9-en`.

[106] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. "Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018, pp. 528–540.

[107] Ankur Parikh et al. "A Decomposable Attention Model for Natural Language Inference". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2016, pp. 2249–2255.

[108] A. Pawar and V. Mago. "Challenging the Boundaries of Unsupervised Learning for Semantic Similarity". In: *IEEE Access* 7 (2019), pp. 16291–16308. ISSN: 2169-3536.

[109] Ted Pedersen et al. "Measures of semantic similarity and relatedness in the biomedical domain". In: *Journal of biomedical informatics* 40.3 (2007), pp. 288–299.

[110] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[111] Matthew E Peters et al. “Deep contextualized word representations”. In: *Proceedings of NAACL-HLT*. 2018, pp. 2227–2237.

[112] Mohammad Taher Pilehvar and Jose Camacho-Collados. “WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 1267–1273.

[113] Mohammad Taher Pilehvar, David Jurgens, and Roberto Navigli. “Align, disambiguate and walk: A unified approach for measuring semantic similarity”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2013, pp. 1341–1351.

[114] Mohammad Taher Pilehvar and Roberto Navigli. “From senses to texts: An all-in-one graph-based approach for measuring semantic similarity”. In: *Artificial Intelligence* 228 (2015), pp. 95–128. ISSN: 0004-3702. DOI: `https://doi.org/10.1016/j.artint.2015.07.005`. URL: `http://www.sciencedirect.com/science/article/pii/S000437021500106X`.

[115] Rong Qu et al. “Computing semantic similarity based on novel models of semantic representation using Wikipedia”. In: *Information Processing & Management* 54.6 (2018), pp. 1002–1021. ISSN: 0306-4573. DOI: `https://doi.org/10.1016/j.ipm.2018.07.002`. URL: `http://www.sciencedirect.com/science/article/pii/S0306457317309226`.

[116] Z. Quan et al. “An Efficient Framework for Sentence Similarity Modeling”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.4 (Apr. 2019), pp. 853–865. ISSN: 2329-9304. DOI: `10.1109/TASLP.2019.2899494`.

[117] Mohiuddin Md Abdul Qudar and Vijay Mago. “TweetBERT: A Pretrained Language Representation Model for Twitter Text Analysis”. In: *arXiv preprint arXiv:2010.11091* (2020).

[118] Roy Rada et al. “Development and application of a metric on semantic nets”. In: *IEEE transactions on systems, man, and cybernetics* 19.1 (1989), pp. 17–30.

[119] Alec Radford et al. “Improving language understanding by generative pre-training”. In: (2018).

[120] Alec Radford et al. “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8 (2019), p. 9.

[121] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *arXiv preprint arXiv:1910.10683* (2019).

[122] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67.

[123] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: `http://arxiv.org/abs/1908.10084`.

[124] Philip Resnik. “Using information content to evaluate semantic similarity in a taxonomy”. In: *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 1*. 1995, pp. 448–453.

[125] M Andrea Rodrıéguez and Max J. Egenhofer. “Determining semantic similarity among entity classes from different ontologies”. In: *IEEE transactions on knowledge and data engineering* 15.2 (2003), pp. 442–456.

[126] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. "A primer in bertology: What we know about how bert works". In: *Transactions of the Association for Computational Linguistics* 8 (2020), pp. 842–866.

[127] Peter J Rousseeuw. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.

[128] Terry Ruas, William Grosky, and Akiko Aizawa. "Multi-sense embeddings through a word sense disambiguation process". In: *Expert Systems with Applications* 136 (2019), pp. 288–303. ISSN: 0957-4174. DOI: `https://doi.org/10.1016/j.eswa.2019.06.026`. URL: `http://www.sciencedirect.com/science/article/pii/S0957417419304269`.

[129] Herbert Rubenstein and John B Goodenough. "Contextual correlates of synonymy". In: *Communications of the ACM* 8.10 (1965), pp. 627–633.

[130] David Sánchez and Montserrat Batet. "A semantic similarity method based on information content exploiting multiple ontologies". In: *Expert Systems with Applications* 40.4 (2013), pp. 1393–1399. ISSN: 0957-4174. DOI: `https://doi.org/10.1016/j.eswa.2012.08.049`. URL: `http://www.sciencedirect.com/science/article/pii/S095741741201010X`.

[131] David Sánchez, Montserrat Batet, and David Isern. "Ontology-based information content computation". In: *Knowledge-based systems* 24.2 (2011), pp. 297–303.

[132] David Sánchez et al. "Ontology-based semantic similarity: A new feature-based approach". In: *Expert Systems with Applications* 39.9 (2012), pp. 7718–7728. ISSN: 0957-4174. DOI: `https://doi.org/10.1016/j.eswa.2012.01.082`. URL: `http://www.sciencedirect.com/science/article/pii/S0957417412000954`.

[133] Victor Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *arXiv preprint arXiv:1910.01108* (2019).

[134] Frane Šarić et al. “Takelab: Systems for measuring semantic text similarity”. In: ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics– Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. 2012, pp. 441–448.

[135] Tobias Schnabel et al. “Evaluation methods for unsupervised word embeddings”. In: *Proceedings of the 2015 conference on empirical methods in natural language processing*. 2015, pp. 298–307.

[136] Kevin Schoepp. “The state of course learning outcomes at leading universities”. In: *Studies in Higher Education* 44.4 (2019), pp. 615–627.

[137] Aliaksei Severyn and Alessandro Moschitti. “Structural relationships for large-scale learning of answer re-ranking”. In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 2012, pp. 741–750.

[138] Aliaksei Severyn, Massimo Nicosia, and Alessandro Moschitti. “Learning semantic textual similarity with structural representations”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 2013, pp. 714–718.

[139] Yang Shao. “HCTI at SemEval-2017 Task 1: Use Convolutional Neural Network to evaluate semantic textual similarity”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 2017, pp. 130–133.

[140] Matthew Shardlow and Raheel Nawaz. “Neural Text Simplification of Clinical Letters with a Domain Specific Phrase Table”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 380–389.

[141] John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[142] Punardeep Sikka et al. “A Survey on Text Simplification”. In: *arXiv preprint arXiv:2008.08612* (2020).

[143] Carina Silberer and Mirella Lapata. “Learning grounded meaning representations with autoencoders”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2014, pp. 721–732.

[144] Roberta A. Sinoara et al. “Knowledge-enhanced document embeddings for text classification”. In: *Knowledge-Based Systems* 163 (2019), pp. 955–971. ISSN: 0950-7051. DOI: `https://doi.org/10.1016/j.knosys.2018.10.026`. URL: `http://www.sciencedirect.com/science/article/pii/S0950705118305124`.

[145] Gizem Soğancıoğlu, Hakime Öztürk, and Arzucan Özgür. “BIOSSES: a semantic sentence similarity estimation system for the biomedical domain”. In: *Bioinformatics* 33.14 (July 2017), pp. i49–i58. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btx238`. eprint: `https://academic.oup.com/bioinformatics/article-pdf/33/14/i49/25157316/btx238.pdf`. URL: `https://doi.org/10.1093/bioinformatics/btx238`.

[146] Emma Strubell, Ananya Ganesh, and Andrew McCallum. “Energy and Policy Considerations for Deep Learning in NLP”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 3645–3650.

[147] Elior Sulem, Omri Abend, and Ari Rappoport. “Semantic Structural Evaluation for Text Simplification”. In: *Proceedings of NAACL-HLT*. 2018, pp. 685–696.

[148] Md Arafat Sultan, Steven Bethard, and Tamara Sumner. “DLS@ CU: Sentence Similarity from Word Alignment”. In: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. 2014, pp. 241–246.

[149] Md Arafat Sultan, Steven Bethard, and Tamara Sumner. “Dls@ cu: Sentence similarity from word alignment and semantic vector composition”. In: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. 2015, pp. 148–153.

[150] Yu Sun et al. “ERNIE 2.0: A Continual Pre-Training Framework for Language Understanding.” In: *AAAI*. 2020, pp. 8968–8975.

[151] Arthur James Swart. “Evaluation of final examination papers in engineering: A case study using Bloom’s Taxonomy”. In: *IEEE Transactions on Education* 53.2 (2009), pp. 257–264.

[152] Kai Sheng Tai, Richard Socher, and Christopher D Manning. “Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2015, pp. 1556–1566.

[153] Jason L Taylor and Dimpal Jain. “The multiple dimensions of transfer: Examining the transfer function in American higher education”. In: *Community College Review* 45.4 (2017), pp. 273–293.

[154] Junfeng Tian et al. “Ecnu at semeval-2017 task 1: Leverage kernel-based traditional nlp features and neural networks to build a universal model for multilingual and

cross-lingual semantic textual similarity". In: *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017).* 2017, pp. 191–197.

[155] Nguyen Huy Tien et al. "Sentence modeling via multiple word embeddings and multi-level comparison for semantic textual similarity". In: *Information Processing & Management* 56.6 (2019), p. 102090. ISSN: 0306-4573. DOI: `https://doi.org/10.1016/j.ipm.2019.102090`. URL: `http://www.sciencedirect.com/science/article/pii/S0306457319301335`.

[156] Julien Tissier, Christophe Gravier, and Amaury Habrard. "Dict2vec: Learning Word Embeddings using Lexical Dictionaries". In: *Conference on Empirical Methods in Natural Language Processing (EMNLP 2017).* 2017, pp. 254–263.

[157] Ashish Vaswani et al. "Attention is All you Need". In: *NIPS*. 2017.

[158] Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. "What is the Jeopardy model? A quasi-synchronous grammar for QA". In: *EMNLP-CoNLL 2007 - Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* June (2007), pp. 22–32.

[159] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. "Sentence similarity learning by lexical decomposition and composition". In: *COLING 2016 - 26th International Conference on Computational Linguistics, Proceedings of COLING 2016: Technical Papers* challenge 2 (2016), pp. 1340–1349. eprint: `1602.07019`.

[160] John Wieting et al. "Beyond BLEU: Training Neural Machine Translation with Semantic Similarity". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* 2019, pp. 4344–4355.

[161] Zhibiao Wu and Martha Palmer. “Verbs semantics and lexical selection”. In: *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics. 1994, pp. 133–138.

[162] Zhilin Yang et al. “Xlnet: Generalized autoregressive pretraining for language understanding”. In: *Advances in neural information processing systems*. 2019, pp. 5753–5763.

[163] Tianyi Zhang et al. “BERTScore: Evaluating Text Generation with BERT”. In: *International Conference on Learning Representations*. 2020. URL: `https://openreview.net/forum?id=SkeHuCVFDr`.

[164] G. Zhu and C. A. Iglesias. “Computing Semantic Similarity of Concepts in Knowledge Graphs”. In: *IEEE Transactions on Knowledge and Data Engineering* 29.1 (Jan. 2017), pp. 72–85. ISSN: 2326-3865. DOI: `10.1109/TKDE.2016.2610428`.

[165] Yukun Zhu et al. “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 19–27.

[166] Will Y Zou et al. “Bilingual word embeddings for phrase-based machine translation”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 2013, pp. 1393–1398.

Appendix E

Resources

Description	**Link**
Inferences from survey conducted in Chapter 2	`https://github.com/Dhivya-C/Survey-on-Semantic-Similarity.git`
The repository of the codebase used in Chapter 3	`https://github.com/Dhivya-C/DSCS-Dataset.git`
The repository of the codebase used in Chapter 4	`https://github.com/Dhivya-C/Transfer-credit-assessment.git`
Survey Form used to build the benchmark dataset in Chapter 4	`https://forms.gle/uu1ydgkb62qsrCc36`