Semantic Min-hashing

Submitted by

Maryam Baig
19I-1247

Supervised by

Dr. Kifayat Ullah Khan Alizai
Masters of Science (Data Science)

A thesis submitted in partial fulfillment of the requirements for the degree of
Masters of Science (Data Science)
at National University of Computer & Emerging Sciences

Department of Computer Science
National University of Computer & Emerging Sciences

Islamabad, Pakistan.

June 2021

Plagiarism Undertaking

I take full responsibility of the research work conducted during the Masters Thesis titled *Semantic Min-hashing* . I solemnly declare that the research work presented in the thesis is done solely by me with no significant help from any other person; however, small help wherever taken is duly acknowledged. I have also written the complete thesis by myself. Moreover, I have not presented this thesis (or substantially similar research work) or any part of the thesis previously to any other degree awarding institution within Pakistan or abroad.

I understand that the management of National University of Computer and Emerging Sciences has a zero tolerance policy towards plagiarism. Therefore, I as an author of the above-mentioned thesis, solemnly declare that no portion of my thesis has been plagiarized and any material used in the thesis from other sources is properly referenced. Moreover, the thesis does not contain any literal citing of more than 70 words (total) even by giving a reference unless I have the written permission of the publisher to do so. Furthermore, the work presented in the thesis is my own original work and I have positively cited the related work of the other researchers by clearly differentiating my work from their relevant work.

I further understand that if I am found guilty of any form of plagiarism in my thesis work even after my graduation, the University reserves the right to revoke my Masters degree. Moreover, the University will also have the right to publish my name on its website that keeps a record of the students who plagiarized in their thesis work.

_____

Maryam Baig

Date: _____

Author's Declaration

I, Maryam Baig, hereby state that my Masters thesis titled *Semantic Min-hashing* is my own work and it has not been previously submitted by me for taking partial or full credit for the award of any degree at this University or anywhere else in the world. If my statement is found to be incorrect, at any time even after my graduation, the University has the right to revoke my Masters degree.

_____

Maryam Baig

Date: _____

Certificate of Approval

It is certified that the research work presented in this thesis, entitled "Semantic Min-hashing " was conducted by Maryam Baig under the supervision of Dr. Kifayat Ullah Khan Alizai.

No part of this thesis has been submitted anywhere else for any other degree.

This thesis is submitted to the Department of Computer Science in partial fulfillment of the requirements for the degree of Masters of Sciencein Data Science

at the

National University of Computer and Emerging Sciences, Islamabad, Pakistan

June' 2021

Candidate Name: Maryam Baig Signature: _____

Examination Committee:

1. Name: Dr. Ehtisham Zahoor Signature: _____
   Assistant Professor, FAST-NU Islamabad.
2. Name: Dr. Adnan Tariq Signature: _____
   Assistant Professor, FAST-NU Islamabad

Dr. Hammad Majeed _____
Graduate Program Coordinator, National University of Computer and Emerging Sciences, Islamabad, Pakistan.

Dr. Waseem Shahzad _____
Head of the Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan.

Abstract

Measuring the similarity between words, sentences, paragraphs and documents is an important component in various Natural Language Processing tasks such as information retrieval, document clustering, word-sense disambiguation, machine translations and text summarizations etc. Recent approaches, Bert, Elmo, Roberta use neural network in order to obtain embedddings. These models require tremendous amount of data and training time to produce state of the art results. Other semantic similarity measures exploit the knowledge bases such as WordNet to perform their estimations using different approaches like shortest path, measuring syntactial similarity of the information content etc between words or short phrases but not longer sentences. In computer science and data mining, min-hashing is a technique for quickly approximating the Jaccard Similarity Score between two sets. This approach has been widely used in computing lexical similarity of the documents. Our aim is to use min-hashing to capture the semantic score of the extended vectors enriched with information obtained from the WordNet in the original word vectors depending upon different relationships like synonyms, antonyms, hyponyms and hypernyms such that the lexical similarity is increased on the basis of semantic information. These enriched vectors are passed to min-hash algorithm to compute the small representational signatures of the vectors, reducing the curse of dimensionality. The similarity of these signatures is equivalent to the semantic score between the two documents. Our approach gives 64% accuracy on a low resource machine on MRPC and SICK dataset.

Acknowledgements

I would like to thank all the little people who made this possible.

Dedication

This is dedicated to the one I love.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Semantic text similarity is a fundamental challenge in many Natural Language Processing tasks, such as Question Answering in different domain such as biomedical [1], Automatic Machine Translations [2], Automatic Text Summarization [3] etc. Natural Language is a very rich resource. Different words are said to be semantically similar if their meaning is same, however, their lexical structure is not. For example, the word 'Destination' and 'Last stop' technically have same meaning. Similarly, 'Crash' can be used in different contexts like an auto accident can be referred to as car crash, a drop in the Stock Market as market crash or going to party without an invitation as crashing the party, in our daily lives.

This nature of language makes it difficult for machines to process the information due to its inability to consider context. In some researches, similarity and relatedness between concepts are modeled in the form of graphs like WordNet [4], BabelNet [5], MeSH [6] or word embeddings like word2vec [7] that enables natural language processing. Modern deep learning models like Elmo [8], Bert [9] are very popular in the domain of natural language processing. These are pre-trained neural network models that classify text on the basis of word embeddings. These techniques are mostly used in text summarization; however, these can be useful for resolving word-sense disambiguation as well. All of these latest neural network based state of the art techniques require huge amount of data and hours of training over expensive resources to produce reasonable results.

In today's world we have large amount of data generated every second, which requires a scalable approach for quick estimations and analysis. Min-hashing [10] is an elegant technique that quickly estimates the similarity between two sets. This approach has been used in Locality Sensitive Hashing for cross-lingual similarity [11], Graph Summarization [12], Estimating Web document Similarity [13]. It is a highly scalable approach and a good estimator of Jaccard Similarity.

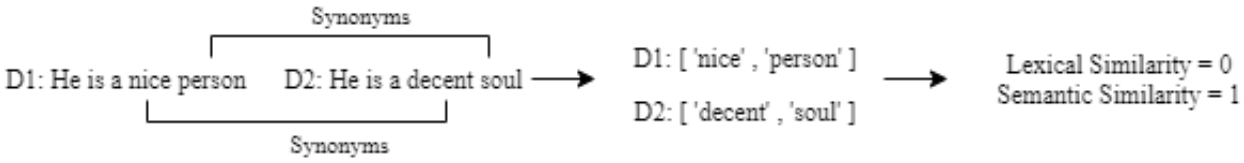

Figure 1.1: Problem Illustration

The motivation is to determine semantic similarity between documents without training on gigabytes of data to obtain decent results. Textual Similarity a.k.a lexical similarity is being efficiently computed through K-shingle tokens and Min-hash algorithm [13] mentioned earlier, however, this technique fails to consider semantics as shown in fig 1.1. In D1 and D2, different words or we can say synonymous words are used to tell the same information that a person is good. If we measure the similarity score through min-hashing with stop words, it approximates to 60% while removing the stop words the score drops to 0, when they are 100% similar semantically. The idea is to add knowledge to the word vectors using a well-structured knowledge base.

We proposed to use min-hash algorithm to compute the semantic similarity of the documents as well. Fig 1.2 shows the high level overview of the proposed solution.We can transform our documents into tokens to form a vector of words. For each token obtain the related words from the knowledge base like WordNet.

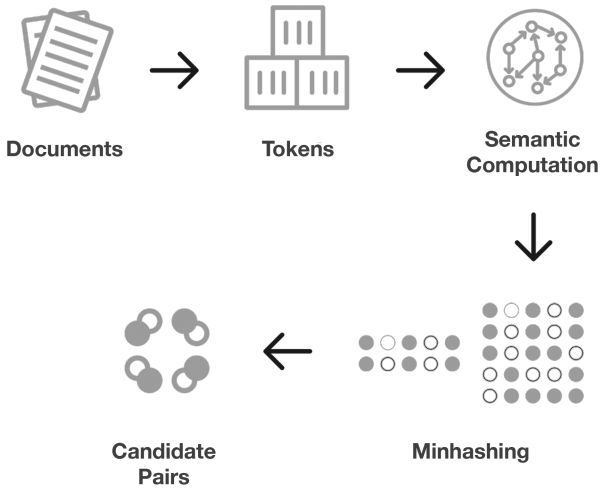

Figure 1.2: High level Overview of the Proposed Solution

Our proposed method models the obtained data such that it preserves the semantic information of a document by increasing the syntactical similarity of the documents

achieved by embedding related concepts in the extending vector representation of the document. These vectors are then encoded in order to pass it to min-hash algorithm. The min-hashing technique produces signatures such that similar words get similar min-hash values, the vectors are converted into codes. These extended encoded vectors are then passed through hash functions to obtain their short signatures. These signatures are meant to preserve the similarity of the documents which is approximately equal to Jaccard Similarity.

Our approach will not only measure the structural similarity of the documents but semantic similarity as well in O(mnk +m^2k) run time complexity, which is the complexity of the min-hash algorithm. The complexity for pre-processing is later discussed in the chapter 4. This method can prove to be helpful in computational linguistic for finding semantically similar items like tweets, articles, as well as clustering semantically similar documents at a large-scale.

1.1 Preliminaries

1.1.1 Knowledge Base

Knowledge bases contains structured semantic information like in WordNet, words are associated with each other through different relationships like synonyms, antonyms, meronyms, hyponyms, hypernyms etc. Fig 1.3 shows the synonyms-antonyms relationship. The words associated with solid arrows are the synonyms and with the dotted line is the antonym.

We have used WordNet as a Knowledge base for concept extraction as it contains all of the required relationships used in this research. WordNet preserve the semantics between words in the form of different relationships like synonyms, antonyms, hyponyms, hypernyms, meronyms.. WordNet contains a total of 155,327 words having multiple senses. It provides part of speech tagging with each word and meaning of each of these words in different senses in the form of synsets including glossaries and descriptions. A synset consists of word text called 'lemma' followed by POS tag as n,v,a,s or r, and its sense number separated by a dot, like bank.n.01, here bank is the lemma, it is noun and the sense is numbered as 1.

Several semantic similarity score methods have been proposed by researchers over the years to estimate the semantic similarity between two concepts by exploiting different factors of the knowledge bases like WordNet, Mesh and BabelNet etc. The basic parameters in these networks are length, depth and density of the network that conserve the semantics in the hierarchy.

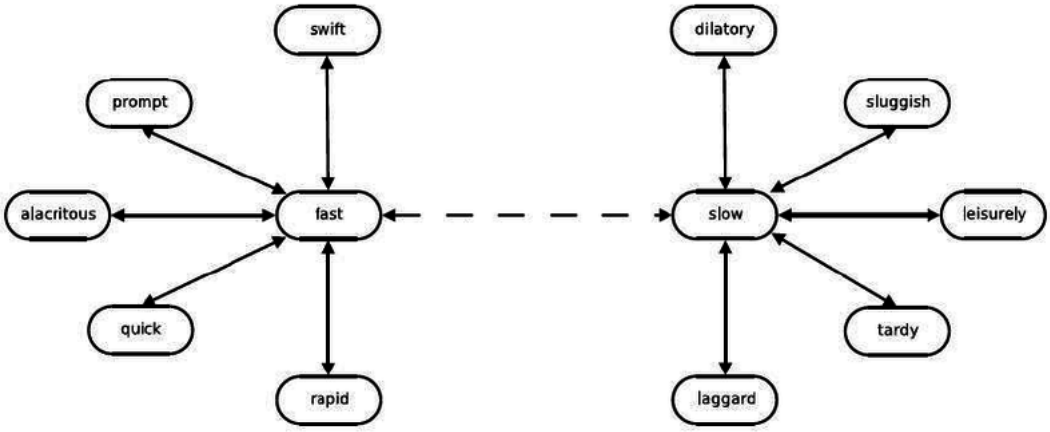

Figure 1.3: A Sample fragment of the semantic network. Solid lines shows the synonyms and dotted line shows the antonyms.

Edge-counting [14, 15], information content [16, 17], neighborhood, description and glosses [18] or the combination of these properties [19] from these networks have been used to estimate the semantic similarity score. The underlying method in the information content, feature based and hybrid approach is the measure of syntactical similarity of semantic information obtained from the network.

Leveraging these knowledge bases in order to enrich the given text, will give us knowledge-aware embeddings through which we can quickly estimate the semantic similarity through *min-hashing* without spending hours on training on large corpora to produce decent results.

1.1.2 Min-hash

Min-hash has been verified to be effective and efficient measure in computing the syntactical similarity in big data systems[10]. Min-hashing Algorithm has been used in Locality Sensitive Hashing to find the nearest neighbors. It represents very large sets into smaller representations while preserving the similarity of the large sets thus reducing the curse of dimensionality. Figure 1.4 shows how min-hash algorithm reduces the dimension of the data and hence performs estimations very quickly.

Due to large number of the permutations k (any arbitrary number), one can estimate the occurred probability of the case to detect the resemblance. It is worth noting that min-hashing works well with binary data [20]. It has been used to solve the real-world problems for compressing social networks [21, 22], search over encrypted data

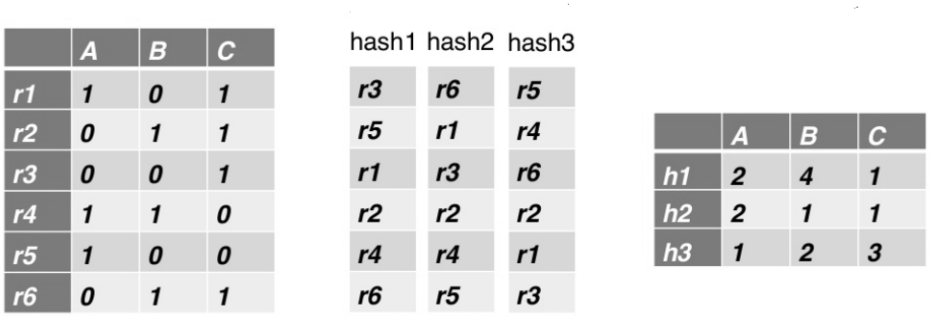

Figure 1.4: Overview of Min-hash Algorithm

[23], information management[24], Graph Summarization [25]. However, none of the proposed solutions have addressed the semantics in min-hashing.

Chapter 2

Literature Review

This section includes some of the closely related work done in semantic computations and minhash algorithm. There has been extensive work on both but this report will include the most related papers focusing on semantics similarity computations using knowledge bases and variants of min-hashing and its applications.

2.1 Related Work on Min-hashing

Min-hashing is frequently used in finding similar items in large data sets for estimating the Jaccard Similarity between two sets. It represents large sets into small signatures while preserving the similarity among the entities. The probability of two sets to have same min hash value is approximately equal to the Jaccard Similarity (Intersection over Union) of the sets[10]. Since it is very expensive to generate random permutations over a large set explicitly, universal hashing is used on indexes to produce permutations.

2.1.1 Variants of Min-hashing

Many variations have been introduced in Min-hashing to improve its efficiency and effectiveness. Table 2.1 shows some variants. Standard Min-hash considers all the elements equally while weighted min-hash associates a probability to each element according to its importance [26]. The added weights increase the bias of items in the sets, thus improving the accuracy of the score.

b-Bit min-hash limits the size of the fingerprint to b-bit, b can be any arbitrary number [27]. The idea is to reduce the space complexity by limiting the size of the fingerprint. Each fingerprint represents an item in the set, a.k.a hash value. However, this

techniques limits the number of bits to be used therefore, it may miss some elements in the sets if there hash value does not occur in b-bits or may overide an exisiting element.

One-permutation min-hash [28] limits the number of permutations to 1 considering that hash value will not collide, which is very unlikely and it only depends on the type of data being used.

None of these variants consider the semantics while estimating the similarity. These techniques are useful in finding similar items but the semantics have never been considered in these approaches, as they only focus on the performance of the min-hash algorithm.

Table 2.1: Some variants of Min-hash Algorithm

Min-hashing Variants	**Approach**	**Goals**	**Journ/Conf**	**Year**
b-Bit Min-hash [15]	Stores only b-bits of each hashed value	Reduce storage space	WWW	2010
One-permutation Min-hash[28]	Limits the number of permutation to 1	Reduce computation cost	ICML	2014
Weighted Min-hash[29]	Associates a probability to each element	Improves accuracy	NIPS	2016

2.1.2 Applications of Min-hashing

These variants have been used in finding similar documents [10]. The documents are represented in the for of sets. Since min-hash approximates the Jaccard Similarity of the sets. We can easily compute the lexical similarity of the documents using min-hash algorithm. Note that the score obtained through min-hashing is actually the Jaccard Similarity of the sets.

Min-hash has been used for measuring cross-lingual similarity in [11]. Wikipedia Corpus is available in multiple languages. In order to obtain the cross-lingual similar document, the association of the multi-lingual documents has been exploited in this research to compute the cross-lingual similar documents.

In graph summarization [12], min-hash can be used for clustering together similar nodes. The research used the increased representation method to introduce the bias for better reults

Web documents Similarity has been computed using K-shingle tokens with min-hashing [13]. This approach tokenizes the documents into K-shingles so that the structurally similar documents can be grouped together.

2.2 Related Work on Semantic Similarity

2.2.1 Knowledge-base Appraoches

There are several researches on estimating semantic similarity score that depends on the knowledge bases and ontologies like, WordNet[1] [4], MeSH[2] [6], BabelNet[3] [5], SENSUS[4] [30], Wikipedia [5]. These knowledge bases are structured in a hierarchical manner while preserving the relatedness of the concepts and words such that similar words are grouped together as shown in Fig 1.3. The length, depth and density are the major factors the can be translated into semantics for similar terms. Based on the factors semantic measures can be categorized as; Structure-based measures, Information content measures, Feature-based measures and Hybrid measures. These measures give the semantic scores between two concepts or words.

Structure based Approach

Structure based measures [31, 6, 14] compute the similarity by measuring the length of the path linking the terms in the knowledge base. Structure based methods are limited to the single ontology as it depends on the 'is-a' relationship in the taxonomy while the distance of two similar words may not be the same in the network.

Information Content Approach

Information content methods measure the differences in the information of the terms by computing the probability of the occurrences of the terms in the corpus [16, 17]. This is dependent on ontology and corpus both to compute the score, thus computationally very expensive.

Feature based Approach

Feature based methods [18] use the properties or relationship of the terms obtained from knowledge bases to compute the similarity. Two terms are said to be more similar to each other if they have more similar features. The features include the descriptions of words, synonym sets, neighborhood information etc.

[1] https://wordnet.princeton.edu/
[2] https://www.ncbi.nlm.nih.gov/mesh/
[3] https://babelnet.org/
[4] https://www.isi.edu/natural-language/resources/sensus.html
[5] https://dumps.wikimedia.org/

Table 2.2: Existing work on Semantic Similarity

Author	Key Idea	Knowledge Base	Corpus Base
R. Rada et al., R.Richardson et al.	Measures the shortest path	✓	
Philip Resnik et al., Jay J. Jianget al.	Requires corpus with ontology to calculate semantic score	✓	✓
Euripides G.M. Petrakis et al.	Takes into account the glossaries and descriptions of the words to produce a score.	✓	✓
Zili Zhou et al.	Relies on weights and relative depth of the knowledge graph	✓	
Masumeh Islami Nasab et al.	Calculates sentence similarity based on synsets obtained from WordNet and give scores acco-rding the relationship between then, synonyms are scored 1.	✓	
MatthewE. Peters et al.	Integratesmultiple knowledge bases like WordNet and a subset of Wikipedia into BERT˜ resulting in a knowledge enhanced repres-entation of word embeddings.	✓	✓
Bishan Yang et al.	Integratebackground knowledge with information from the currently processed text andemploys an attention mechanism with sentinel to adaptively decide whether to attend to background knowledge and which information from Knowledge base is useful.	✓	✓
J. Devlin et al.	Pre-training of deep bi-directional transformers for language under-standing		✓
T. Mikolov et al.	Estimates word representation in vector space using cosine similarity		✓

Hybrid Approach

Hybrid approaches used the combination of the above mentioned techniques to compute the similarity in terms. However, for computing similarity across ontologies, feature based methods can prove to be useful.

These measures requires a lot of information processing to give a good similarity score. Table 2.2 summarizes the pioneer work in semantic computations along with their limitations. All these methods except structure based ultimately compute the lexical similarity of the content obtained from different properties, making these approaches computationally expensive for semantic similarity measurement.

[32] proposed an ontology based approach to determine the semantic similarity of the web documents by obtaining the relations from the semantic network for each keyword, in the form of the graph. They take the union of the graphical formation of the document to compute the semantic similarity score. However, this approach is not exhaustive and scalable.

[33] measures the semantically similar scientific articles by considering only the title, abstract and keywords in the articles. This approach first calculates sentence similarity by obtaining the sysnets of the words from WordNet, if two words are synonyms they are scored one, if the relation between the words is 'is-a' the score is 0.95 and for 'part-of' relation the score is 0.85. After getting the sentence scores for title, abstract and keywords, it calculates the weighted mean to produce a semantic similarity score between two documents. This approach limits the use of information and is computationally very expensive.

2.2.2 Corpus-base Approaches

These approaches used textual corpus in large amounts to draw rules in the form of embeddings by representing words in numbers using cosine similarity or other measures.

Non-Contextual Embeddings

Recently word embeddings and autoencoders have been in the limelight. It represents document in the form of vectors and their are a number of techniques, few of which are Word2vec [34], Glove [35], fastText [36]. Word2Vec's [34] embeddings helps to improve majority of the NLP tasks but subword infromation is not captured. FastText [36] uses the word2vec model and improves its efficiency and performance, also helps to captures the subword context. However, Glove [35] aims to bring word prediction

algorithms and word statistics together over the entire corpus. It considers the corpus's co-occurrence statistics as well as the effectiveness of prediction-based approaches.

These techniques are not contextual, means they do not take into account the context in which the word is used, hence loosing the semantics. ELMo [8] introduced the contextualized embedding capturing semantics with context. Bert [37] uses autoencoders for text representation in the form of vectors.

Contextual Embeddings

Contextual word representations, which are generally trained on unstructured, unlabeled text, lacks clear linkage to real-world items and are frequently incapable of remembering information about such entities. Knowledge bases (KBs) are a rich supply of human-curated, high-quality data that may be utilised to base above mentioned models. Furthermore, they frequently incorporate information that is not available in raw text, and they can encapsulate factual knowledge that is hard to learn through selection preferences because of the rare occurrences of common sense knowledge or long-term dependencies.

Knowledge-aware Embeddings

KnowBert [38] is a generic and efficient model for incorporating prior knowledge into a deep neural network. It trains entity linkers by self-supervision on unlabeled data, resulting in general-purpose knowledge-enhanced representations that may be used for a variety of downstream applications.

A survey on document clustering on the basis of semantic similarity has shown that most of the methods proposed have used term frequency–inverse document frequency or WordNet as the knowledge base to obtain semantic information and then used K-mean clustering and some of its variants to cluster documents [39]. None of the techniques involved the min-hashing technique to compute the semantic similarity. It is worth noting that min-hashing can also be used in clustering.

Semantic networks contain a lot of information on the similarity and relatedness among different concepts. It is worth mentioning here that machine learning and deep learning models are also being used for word-sense disambiguation but these models require extensive pre-training on a large corpus to give meaningful results. Since data is growing very fast, we need a scalable approach that can efficiently approximate the similarity score while preserving the semantic information. Jaccard Similarity is a good estimator for finding similar sets. Min-hash quickly estimates the Jaccard Similarity between two sets. As, minhash works on the sets and word embeddings are in the

form of vectors, we can use minhash to estimate similarity among the two pair of sentences.

The objective of this research is, to provide a scalable solution to find semantically similar documents by exploiting the relationships of words in the taxonomy. The associations between words, like 'is-a', 'part-of', 'has-a' provides a bases to explore the semantics. Min-hash provides the advantage of scalability as each instance is reduced to a fixed size signature.

Chapter 3

Methodology

This section discusses the proposed solution shown in Fig 1.2 in detail, which aims to capture the semantically similar candidate pairs through min-hashing. The knowledge bases like WordNet provides the semantic information in the form of synonyms, antonyms, meronyms etc. These relationships between the words can be modeled in such a way that it can increase the bias, which in turn captures the semantics of the documents.

The objective is to provide a scalable solution to find semantically similar documents by exploiting the relationships of words in the taxonomy. The associations between words, like 'is-a', 'part-of', 'has-a' provides a bases to explore the semantics. Min-hash provides the advantage of scalability as each instance is reduced to a fixed size signature. Following section discusses the proposed approach step by step. Fig 3.1 shows the flow chart of the process, which are given below in detail.

3.1 Illustration

Min-hashing is widely used in computing syntactical similarity of the documents in information management systems [18], searching in encrypted data [23] and in graph summarization [12]. By leveraging knowledge bases like WordNet we can embed semantic information in the original vector to in order to get the semantic similarity score.

Min-hash takes the data in the form of sets to perform computation. This can be achieved by tokenizing the documents into words. Let's say we have two documents,

d1: ['he', 'is', 'a', 'nice', 'person'] d2: ['he', 'is','a', 'decent', 'soul']

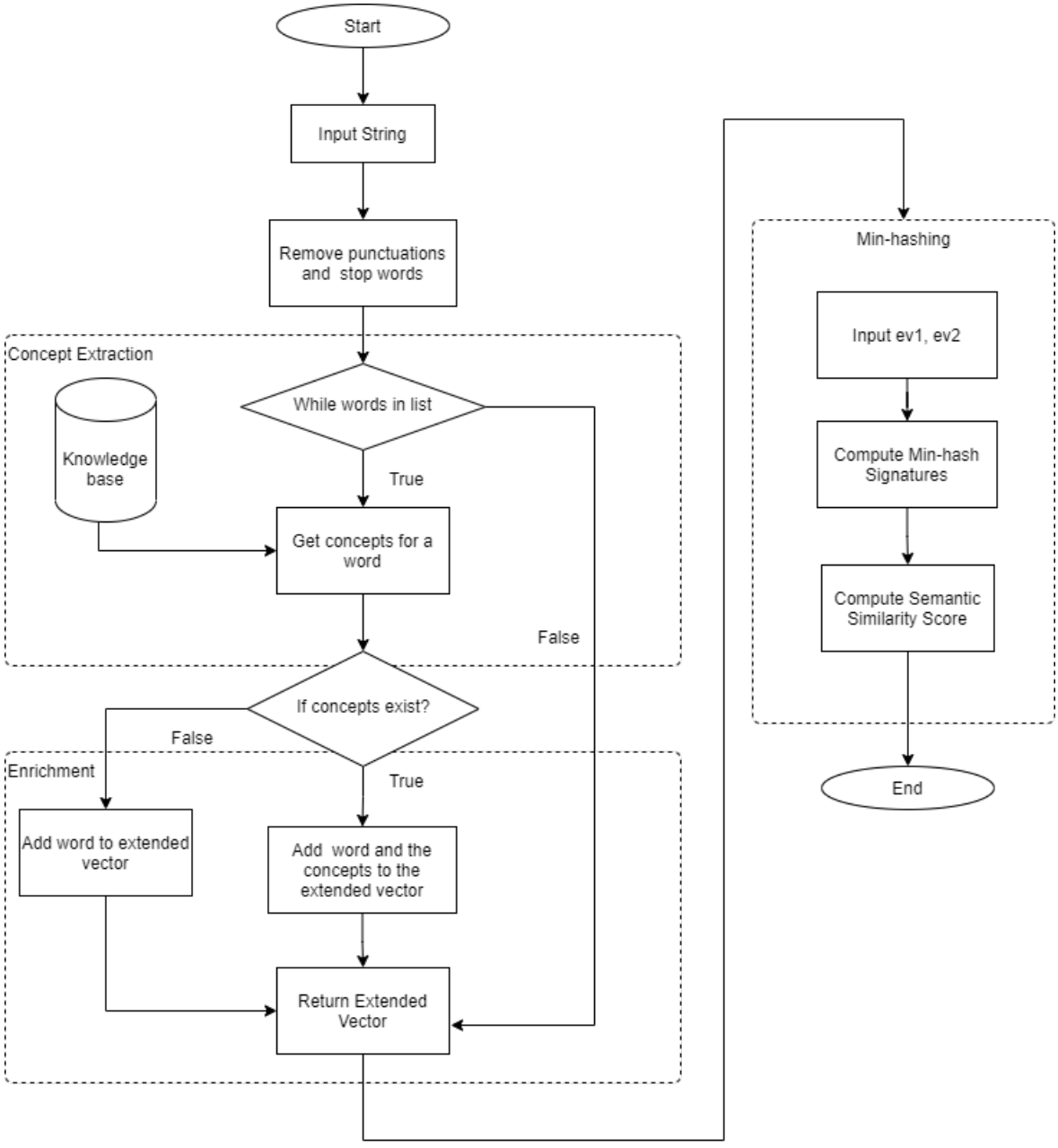

Figure 3.1: Flow of the semantic score computation

After removing the stop words, the vectors will have words that are adding meaning to the sentence.

d1: ['nice','person'] d2: ['decent','soul']

We can see that the documents are similar to each other syntactically approximately 50%, but they are 100% similar semantically, which means both the sentences have same meaning yet their similarity score is very low. For considering the semantics, we propose to obtain closely related words from the semantic network as shown by fig 3.2 and add them to the words vector, given by 3.5 to obtain the extended vector.

This vector is then encoded as a sequence in order to pass to minhash algorithm for computing scores. For example, the word 'nice' has a synonym 'decent', and the word 'decent' has 'nice' as its synonym, we can see that both words occur in the synsets of each other. Similarly, 'person' and 'soul' are synonyms of each other in WordNet.

Adding these words to the words vector will increase the bias of the semantic terms which are adding meaning to a phrase or sentence, causing the collisions in the min-hash in turn resulting in the increased score in terms of semantics.

The process of estimating semantic similarity of the above sentences can be achieved in three steps,

- Concept Extraction
- Enrichment
- Min-hashing

Each of the above steps are discussed below in detail.

3.1.1 Concept Extraction

In this phase, concepts are extracted from a knowledge base in order to enrich feature vectors. Knowledge bases preserve the semantics between words in the form of different relationships like synonyms, antonyms, hyponyms, hypernyms, meronyms, we call these relationship as *concepts*. The feature vector after the pre-processing contains only the meaningful terms hence removing the bias added due to stop words. The concepts are obtained against each word in the original sentence vector. The number of concepts obtained against each word may vary (can be zero). Fig 3.2 shows the concept extraction phase where related concepts are obtained against each vector.

The focus of this research will be the following relationships,

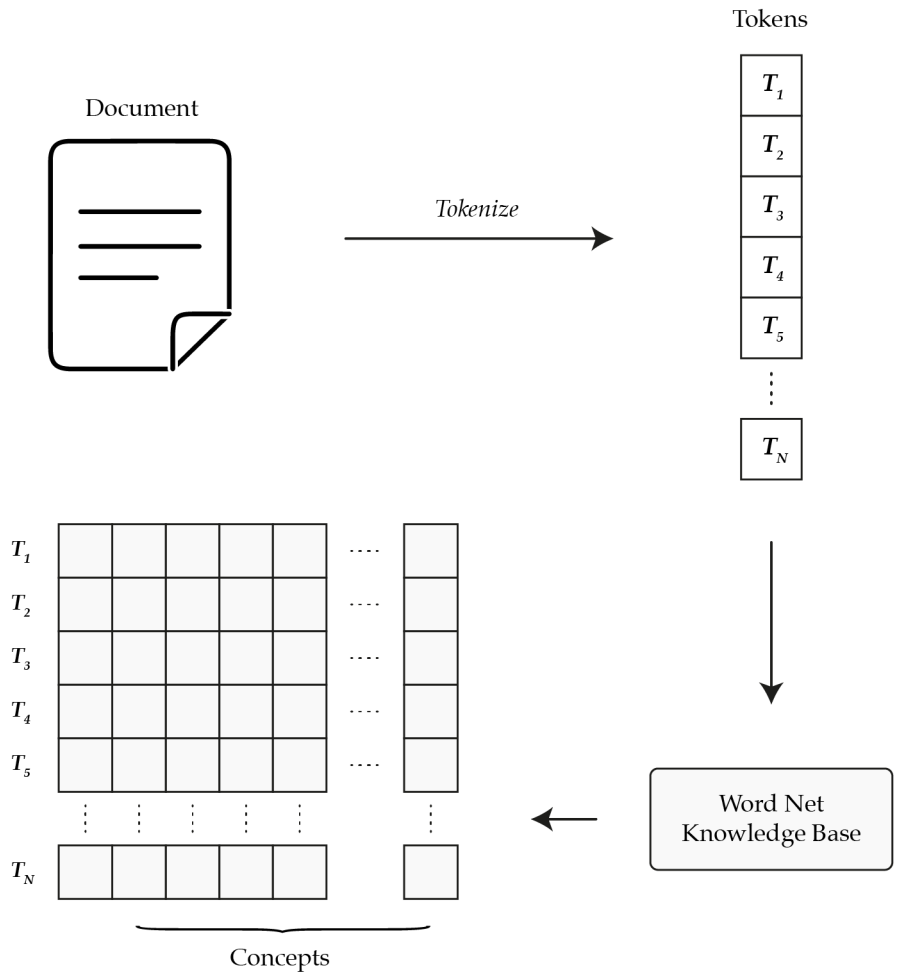

Figure 3.2: Concept Extraction Phase

- **Synonyms**: Two words having same meaning are called synonyms of each other, like big is the synonym of large. WordNet contains synonyms in the form of synsets; list of similar meaning words grouped together.
- **Antonyms**: Two words having opposite meaning are called antonyms, like big is antonym of small. Some words may or may not have opposite words.
- **Hyponyms**: The words having a specific example of a general word are hyponyms of each other for example, color is a general word, red is a specific color so red is the hyponym of color. If we consider the WordNet as a tree, the leaves from each node are the specific terms of its general form at the node. for the purpose of this research we have considered one hop away hyponyms as there is very less chance of concepts collision if we go down deeper to specific words.
- **Hypernyms**: Hypernyms are the general form of a specific word, opposite to the hyponyms. For examples, color is the hypernym of the red as it represents a broader meaning as in fig 3.3. So if different colors are used in a sentence, we can add color for each of these words increasing the semantic similarity. For the purpose of this research we have considered concepts up to three hops away hypernyms as there is a chance of getting common words for two closely related words.

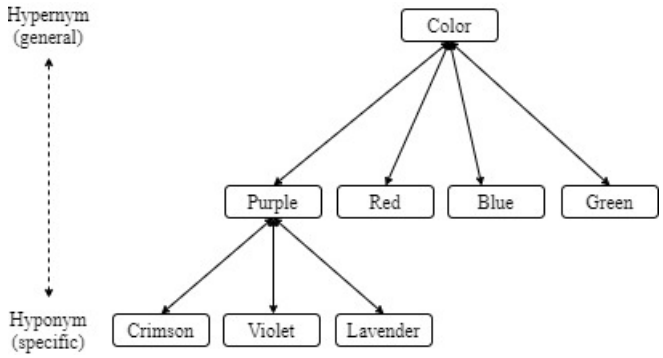

Figure 3.3: Hyponym-Hypernym Relationship

WordNet is very rich in synonyms, evident from the Fig 3.4. The number of concepts obtained for each relation against each token has been counted to analyze the resourcefulness of the WordNet. It gives significantly high average of 115 for synonyms while antonyms are very few. In addition to antonyms, number of hyponyms and hypernyms are also insignificant as compared to synonyms.

Each of the above relations mentioned is independent from the other, therefore, in order to make a meaningful enriched word vectors only one relationship is considered

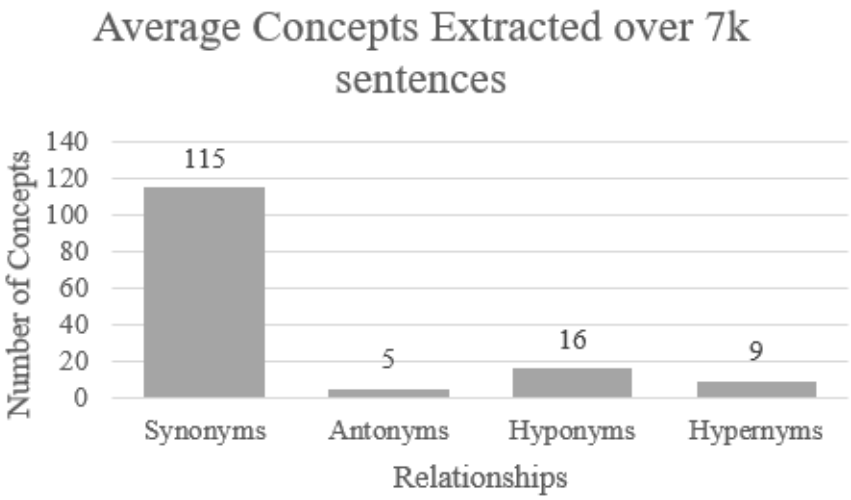

Figure 3.4: Average number of concepts obtained for each relation

at a time. In the example below, we obtained synonyms from the extraction process of the words, 'decent','person','nice','soul'. Extracting concepts for 'person' and 'soul' will give

person: ['individual', 'somebody', 'someone', 'mortal', 'person', 'soul']

soul: ['individual', 'somebody', 'someone', 'psyche', 'mortal', 'person', 'soulfulness', 'soul']

We can see that few synonyms for both the words person and soul are common. Similarly, obtain the concepts for 'decent' and 'nice'.

3.1.2 Vector Enrichment

This phase is responsible for the formation of the extended vectors. These vectors are obtained by the addition of extracted terms into the original vector such that the representation of each token is increased by its related terms. This is given by,

$$\sum_{i=1}^{N} T_i + C_t \tag{3.1}$$

where C is the Concept for Tokens T from 1 to N (N is the number of tokens per document and t is the size of the concept vector). In order to remain in the context,

a threshold value of t=3 is determined for enrichment which means that we will use first three values of the extracted concepts for each word resulting in enriched vectors, called *extended vectors*.

If two words are related to each other, the extracted word set must have at least one word similar, hence causing the collision. The process is demonstrated in the fig 3.5. The output of this phase is an encoded vector which is encoded using a very simple approach i.e. each word is given a number in a sequence starting from 1, where similar words get the same identification number.

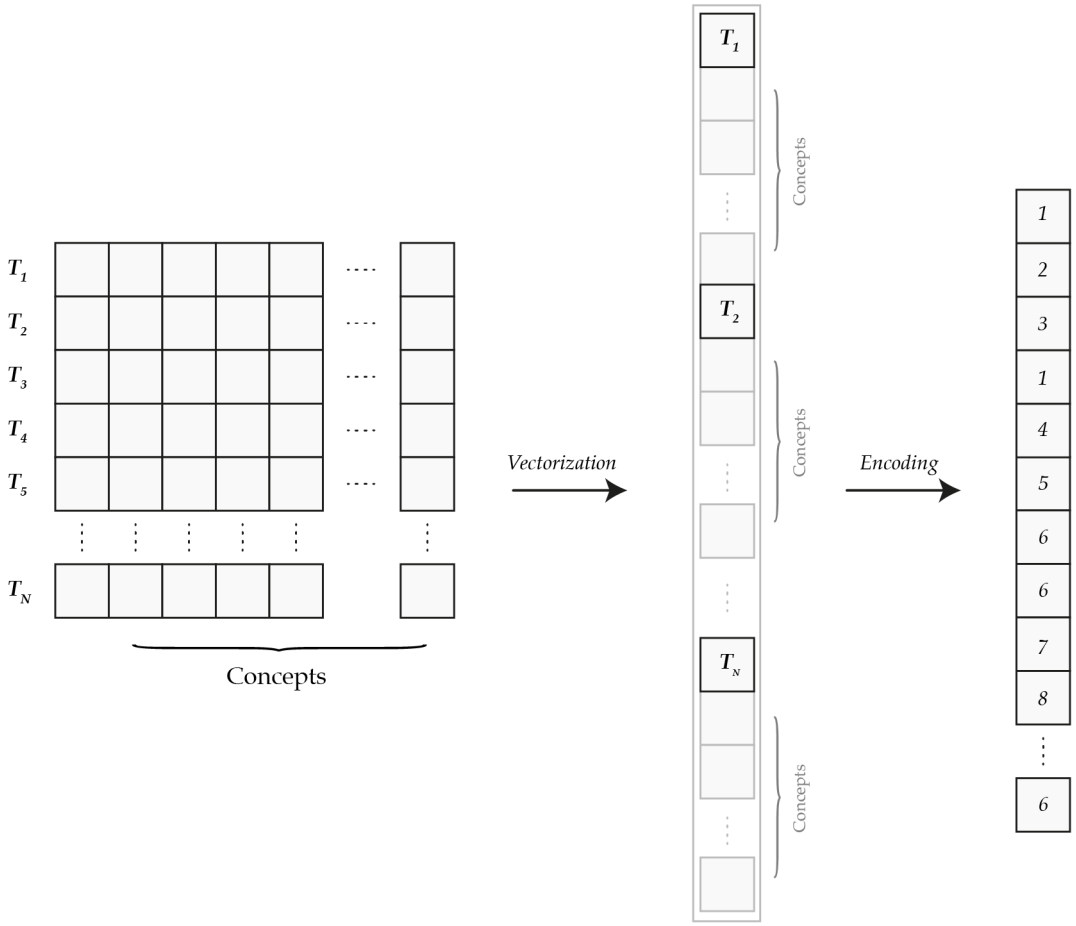

Figure 3.5: Enrichment Phase

The formation of the extended vectors depend on one of the relationships at a time. Using the synonyms obtained from extraction phase, resulting extended vectors are,

ev1=['nice', 'good', 'decent', 'gracious', 'person', 'individual', 'somebody', 'someone']

ev2=['decent', 'good', 'nice', 'dainty', 'soul', 'individual', 'somebody', 'someone']

The extended vectors are now highly similar as each of the words are present in both the vectors, increasing the representation of the similar words. The size of the extended vectors adds to the space complexity by a factor of t. The size of the extended vector is given by,

$$size(ev) = n + t * n \tag{3.2}$$

where n is the size of the original vector and t is the threshold which determines the number of concepts to be added to the extended vector. Fig 3.6 shows the average increase in extended vector for the relationships used in this research.

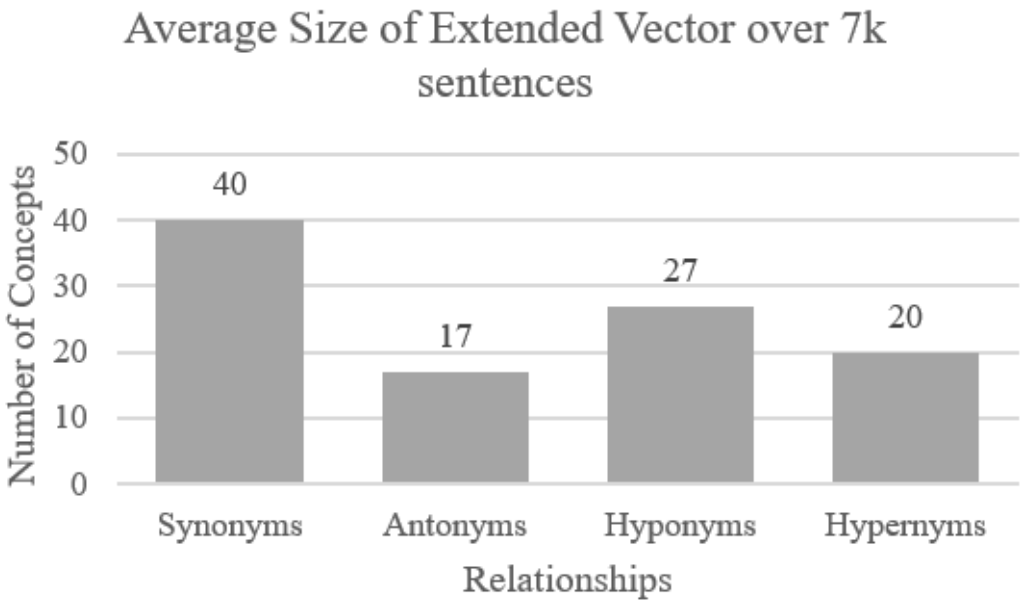

Figure 3.6: Average size of extended vectors after vectorization

The size of the extended vector is directly proportional to the threshold, given by equation 3.3. Greater the threshold, greater is the size of the extended vector and vice versa.

$$size(ev) \propto t \tag{3.3}$$

where ev is the extended vector and t is the threshold which determines the number of concepts used for each token. In Fig 3.6 synonyms gives the maximum size of the extended vectors when threshold is 3, where hypernyms are half the size of the synonyms, antonyms resulting in the smallest vector size.

3.1.3 Min-hashing

Min-hash algorithm represents large sets into smaller representations through hashing, hence significantly reducing the size of the sets to be compared quickly while preserving the similarity of the sets. Leveraging this property of min-hash in order to reduce the size of the extended vectors and obtaining equal sized encoded vectors where size of the vectors depends on the number of hash functions used.

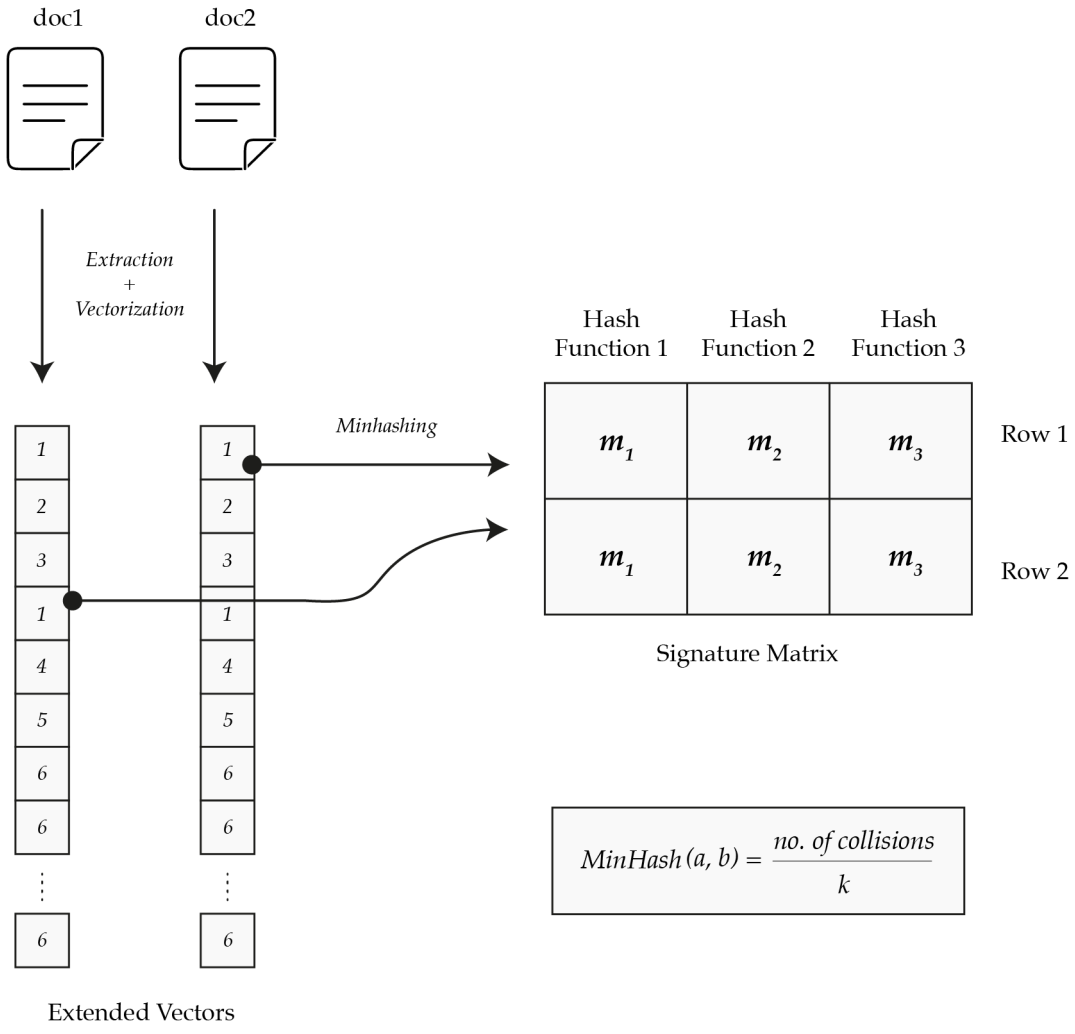

Figure 3.7: Min-hashing

The k length vectors obtained from min-hash algorithm are then compared to cal-

culate the similarity score, which is the given by formula,

$$minhash\_similarity\_score(a,b) = \frac{number\ of\ collisions}{k} \tag{3.4}$$

where a and b are sets of integer obtained from min-hashing and k is an arbitrary number of hash functions used in calculating the signatures. This score is approximately equal to the Jaccard Similarity of the semantic set ranging between 0 and 1.

The extended vectors ev1 and ev2 obtained in the previous section give the semnatic similarity score of 0.8 which shows that the two sentences are highly similar despite the structure of the sentences make them totally different from each other.

3.2 Implementation

The proposed algorithm to compute the semantics, takes the word vector v as an input. This vector is obtained from tokenizing the document shown in fig 3.2 and removing the stop words. The threshold to use the minimum number of related words is set to 3. It has been observed that as you go deeper into the concepts, the meaning of the words change, this threshold helps in maintaining the concepts list to closely related words. Also, if the context is different then the concepts obtained for a particular token, the enrichment will not effect the resulting score drastically.

Algorithm 1 shows the steps to form the extended vectors which are then fed to min-hash algorithm for scores.

Algorithm 1 Semantic Computation

```
Data: v: [w1,w2,...wk]
Result: ev: [w1,w1s1,w1s2,w1s3...wk,wks1,wks2,wks3]
Initialization
 threshold ←3
 ev←emptyList
 while word in v do
    concepts ← ConceptExtraction(word)
     if concepts then
        ev.append(word)
         ev.append(concepts[ : threshold - 1])
    else
        ev.append(word)
    end
end
```

Initialize an empty list for the extended vector (ev). For each word w in the vector v, find the cooncepts from the WordNet. If the list of concepts is not empty, append the terms till the threshold value which is set to 3 in our case, otherwise just add the word to ev and move to next word. This function will return a vector ev which contains the words and their related words for a document. These relationships can be synonyms, antonyms, hyponyms or hypernyms. We will only use one relationship at a time to see the effects of each one on them on the semantic similarity score.

Complexity

The time complexity of this algorithm is O(n) where n is the number of words in the word vectors. However, WordNet lookup is binary search, a runtime cost of O(log n) is added. So the complexity of the formation of extended vectors turn out to be,

$$O(n\ logn) \tag{3.5}$$

Min-hash reduces the space complexity from n to k, where k is the number of hash functions used. For larger documents a good value of k may range from 50-100.

Chapter 4

Evaluations

This chapter discusses the experimentation details and results obtained after applying the proposed methodology. First section contains the details of the datasets used for experimentation followed by the experimentation section which contains the implementation details and statistical analysis.

4.1 Dataset

The proposed approach is tested on two very popular datasets given below,

4.1.1 Microsoft Research Paraphrase Coprus

Microsoft Research Paraphrase Coprus (MRPC) [40] consists of 5800 pairs of sentences in English language, annotated by humans. The sentences are paraphrased, which means the pair of sentences have same meaning but their syntactical structure is different. The pairs are annotated in terms of quality. A quality score of 1 is given to the pair having good paraphrasing while 0 is given to low quality pairs.

4.1.2 Sentence Involving Compositional Knowledge

Sentence Involving Compositional Knowledge (SICK) [41] dataset consists of 10,000 pair of sentences in English language. Each pair of sentence is annotated for relatedness in meaning giving a relatedness score which ranges between 1-5. This score has been normalized using a threshold of 2.5. Values greater than this threshold are classified as 1 and 0 otherwise. It also contains the annotations of entailment which

are out of the scope of this report. We have used the relatedness score for obtaining the performance statistics for our approach.

4.2 Experimentation details

This section includes the details of environment and packages that are used in implementing the proposed method. It is followed by evaluation section which comprises of hypothesis testing and statistical analysis of the algorithm in terms of accuracy score and other metrics.

4.2.1 Environment Setup

Hardware

The hardware used is standard PC with Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz processor and 16GB RAM running 64-bit Windows 10 for the development and testing of the proposed approach. However, this implementation could be easily run on a low resource machine for comparing two documents.

Softwares

The proposed method is implemented using Natural Language Processing Toolkit (NLTK)[1] in Python 3[2]. We have used ntlk corpus for WordNet for concept extraction phase. Other than this regex tokenizer, stop words and n-grams has been used from NLTK. Pandas package is used reading and processing data.

Evaluations has been performed using the metrics library from sklearn[3]. The evaluation metrics include the accuracy score as along with complete classification report comprising of precision, recall and F1 score as well as sensitivity and specificity. The scores ranges from 0-1 are classified into classes 0 and 1 using a threshold of 0.5.

[1]https://www.nltk.org/
[2]https://www.python.org/downloads/
[3]https://scikit-learn.org/stable/

4.2.2 Evaluations

Hypothesis

Min-hashing gives a similarity score of the two sets which are enriched with concepts via knowledge base. The similarity score ranges between 0 and 1, where 0 means no similarity and 1 means 100% similar. Our hypothesis is that *"the semantic similarity score should be higher than the lexical similarity score of two pair of sentences which are not similar lexically but semantically"* given by equation 4.1,

$$\textit{Semantic Similarity Score} > \textit{Lexical Similarity Score} \tag{4.1}$$

Since we are using the increased representation method i.e., adding concepts from knowledge base so that collisions are increased by introducing the bias in the form of similar concepts, therefore, the extended vectors will have similarity score higher than non-enriched vectors. For each pair of sentence we observe that this semantic similarity is always higher than the lexical similarity score. This hypothesis 4.1 holds for all the pair of sentences of MRPC and SICK Dataset. Let's look at the statistics with an example from the dataset.

- **Semantically Similar Sample**

  Consider a pair of sentence from MRPC dataset with Quality score as 1 after cleaning the punctuation's,

  Original Sentence:

  "Amrozi accused his brother whom he called the witness of deliberately distorting his evidence"

  Paraphrased Sentence:

  "Referring to him as only the witness Amrozi accused his brother of deliberately distorting his evidence "

  These two sentences differ only in positioning of words while most of the words used in these sentences are same expect for 'referring' after removal of the stop words.

  After applying the proposed method we obtain a similarity score of 0.75 using synonyms which is close to 1 as compared to the lexical similarity which is 0.3. Note that min-hash is an estimator of Jaccard Similarity since the two scores are

Table 4.1: Similarity Scores Comparison for Sample 1 where Quality=1

Type of Similarity	Measures	Scores
Lexical	Jaccard	0.24
	Min-hash	0.3
Semantic Min-hash	Synonyms	0.75
	Antonyms	0.66
	Hyponyms	0.5
	Hypernyms	0.4

very close to each other. Scores for each relationship used for enrichment with semantic concepts are given in 4.1, which shows that our hypothesis 4.1 holds. It can be seen that the synonyms enrichment give a better score with min-hash where the number of hash functions used are 7 in this example.

- **Semantically Dissimilar Sample**

  Consider another pair of sentence with quality score 0 i.e. the sentences are not similar semantically.

  Original Sentence:

  "Yucaipa owned Dominicks before selling the chain to safeway in 1998 for 25 billion"

  Paraphrased Sentence:

  "Yucaipa bought Dominicks in 1995 for 693 million and sold it to safeway for 18 billion in 1998"

Table 4.2: Similarity Scores Comparison for Sample 2 where Quality=0

Type of Similarity	Measures	Scores
Lexical	Jaccard	0.0
	Min-hash	0.0
Semantic Min-hash	Synonyms	0.08
	Antonyms	0.08
	Hyponyms	0.25
	Hypernyms	0.33

From the table 4.2, we can see that there is no lexical similarity between the two sentences, and semantic similarity score is very close to zero for Synonyms and Antonyms, where as Hyponyms and Hypernyms does not give very promising

results. The hypernyms used in both sample 1 and sample 2 are 1 hop away. However, the hypothesis 4.1 holds in this case as well as the semantic score is higher than the lexical similarity score of the pair of sentences.

Metrics Evaluation

The results are obtained on different levels as different relations are exploited in order to obtain a semantic score. As mentioned in section 3, the focus of this report will be four relations named, synonyms, antonyms, hyponyms and hypernyms, the classification report of each relation is added in the form of a table. For each relation, we have obtained accuracy, precision, recall and F1 score along with specificity and sensitivity. These metrics show the performance of our approach on the listed datasets in section 4.1.

- **Synonyms**

  Synonymous words have similar meanings. WordNet is very rich in synonyms as shown in Fig 3.4. For the sake of simplicity 3 or less concepts are used for each word in the original sentence in enrichment phase. Using this approach we obtained 63% accuracy on MRPC whereas 56% on SICK dataset. The detailed classification report is shown in table 4.2.2. It shows the F1 score of 0.69 on MRPC and 0.66 on SICK for positive class.

Table 4.3: Evaluation on Synonyms

Datasets	Measure	Metrics		
		Precision	Recall	F1 Score
MRPC	Sensitivity	0.79	0.61	0.69
	Specificity	0.45	0.66	0.54
SICK	Sensitivity	0.99	0.50	0.66
	Specificity	0.24	0.96	0.38

F1 score shows that we have relatively low false positives and low false negatives, which means our approach is classifying true positives and true negatives accurately with little error. The precision goes very high up to 0.99 on SICK dataset for synosyms.

- **Antonyms**

  Antonyms are words having opposite meanings. WordNet is not very rich in antonyms as shown in Fig 3.4, on average only 5 antonyms are obtained for over 7k sentences which is significantly small number as compared to synonyms. For

the sake of simplicity 3 or less concepts are used for each word in the original sentence, if obtained. The notion behind using the antonyms is that the two words may have a same opposite meaning word, thus reducing the size of the extended vector. Since the antonyms occurrences are very low, we cannot rely on antonyms for enrichment.

Table 4.4: Evaluation on Antonyms

Datasets	Measure	Metrics		
		Precision	Recall	F1 Score
MRPC	Sensitivity	0.78	0.62	0.69
	Specificity	0.45	0.65	0.53
SICK	Sensitivity	0.99	0.50	0.66
	Specificity	0.24	0.96	0.38

However, Antonyms give the accuracy score of 55% on SICK and 63% on MRPC data. Table 4.2.2 shows the statistics obtained after enrichment with antonyms containing precision, recall and F1 score. The scores show that there is not much difference between the metrics score of synonyms and antonyms, even when there are very few antonyms obtained on average for a sentence.

- **Hyponyms**

  Hyponymy is a relation between words from general to specific. If we consider WordNet as a graph, then the hyponyms are the leafs of a node. As we move down the graph the leaves contain the more specific terms for example, red is the hyponym of word color. So the notion behind using hyponyms is to use specific terms which are 1 hop away from the node word. Hypnonyms are fairly reasonable in numbers as compared to antonyms but still very less than synonyms.

Table 4.5: Evaluation on Hyponyms

Datasets	Measure	Metrics		
		Precision	Recall	F1 Score
MRPC	Sensitivity	0.76	0.68	0.72
	Specificity	0.46	0.56	0.50
SICK	Sensitivity	0.97	0.56	0.71
	Specificity	0.25	0.91	0.39

For the sake of simplicity 3 or less concepts are used for each word in the original sentence. 61% accuracy is obtained on SICK while 64% on MRPC. Whereas F1 score has increased to 0.72 and 0.71 for MRPC and SICK dataset, respectively as

shown in table 4.2.2. The improved metrics show that hyponymy is a more meaningful relationship in terms of capturing semantics from WordNet as compared to synonyms and antonyms.

- **Hypernyms**

  Hypernymy is the relationship between words from specific to general. It is the opposite of the hyponymy, just like synonyms and antonyms. For, example color is the hypernym of the word red. This relationship has an interesting property, two or more leaves may end up in a same word, hence better chances of collision. Using this relationship, generalization of the terms used in a document can be done in a better way.

Table 4.6: Evaluation on Hypernyms

Datasets	Measure	Metrics		
		Precision	Recall	F1 Score
MRPC	Sensitivity	0.76	0.69	0.72
	Specificity	0.46	0.55	0.50
SICK	Sensitivity	0.98	0.56	0.71
	Specificity	0.25	0.92	0.39

However, the accuracy obtained is 61% on SICK and 64% on MRPC which is similar to the accuracy of the hyponyms. Table 4.2.2 shows the scores of the metrics which are also similar to that of hyponyms.

Since hypernyms have the ability to generalize, we have tested our approach on hypernyms which are 3 hops away, which means that the words which are 3 hops are more general terms. Fig 4.1 shows the accuracy scores for hypernyms up to 3 hops away. It can be seen that there is no significant change in the accuracy. So we can conclude that the words which are 3 hops away from the given word does not have any significant effect on the semantic similarity obtained from our proposed method.

The comparison of the accuracy scores of all the relationships used in this report is shown in table 4.2. It can be seen that the *Hypernyms* give the most consistent result as it has same accuracy score of 64% on both datasets which is also the maximum.

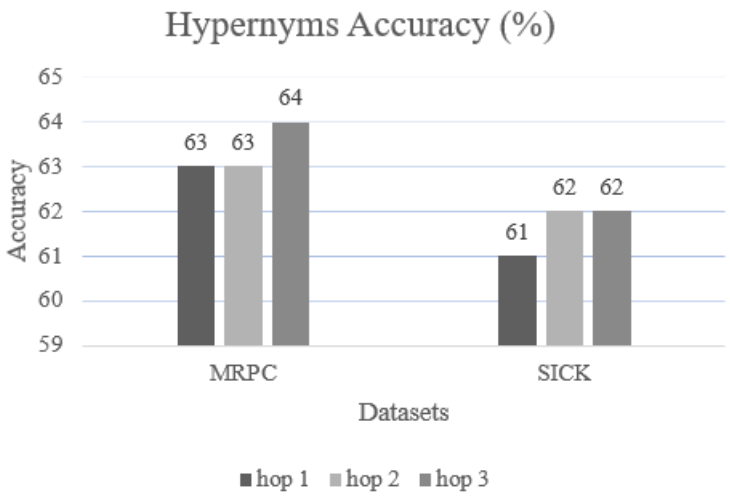

Figure 4.1: Accuracy Scores for Hypernyms up to 3 hops away

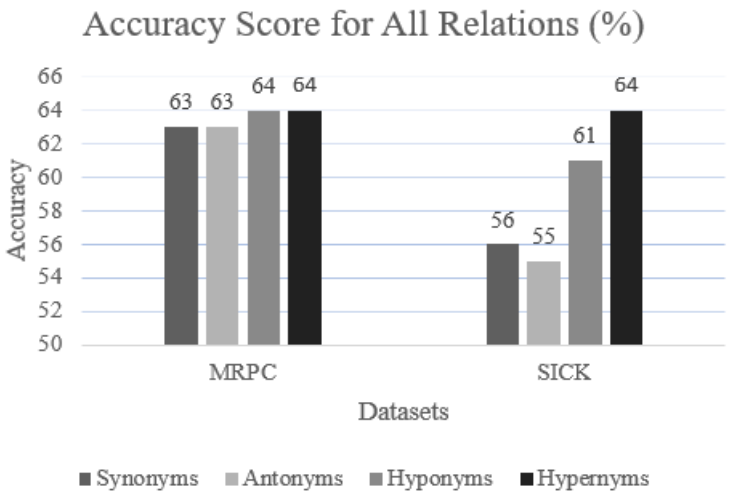

Figure 4.2: Accuracy Scores for All Relationships used for Enrichment

Chapter 5

Conclusion

Semantic Similarity is a very challenging task. There has been always a trade-off of between the accuracy and computational expense. Knowledge based approaches in the past calculated semantic scores between phrases and words on the basis of their distance or information content etc preserved in the form of a graph. These measures did not take into account the holistic view of a document when it comes to measuring semantic similarity. However, this problem was solved by Corpus-based approaches. These approaches require extensive training on very large data up to billions of gigabytes for producing state of the art results.

Since data is increasing at an exponential rate, there is a need of a scalable approach that can be used to estimate similarity among text documents, these documents can be a tweet, an article, a post, a blog etc. Min-hash has been used widely in estimating lexical similarity of the text documents. This technique is very efficient both in terms of time and space complexity when compared to corpus based approaches.

We proposed to enrich document vectors using a Knowledge-base. Concepts are extracted from knowledge bases against each token in the document vector resulting in extended vectors. These extended vectors introduce a bias or weight by increasing the representation of the similar words in the concept vector. Concept vectors of closely related words should contain similar words.

The results show that the proposed approach works well for semantic computation as shown by the results in chapter 4. Out of all the relationships under consideration, hypernyms give the most promising result of 64% which has been proved by experimentation. Our proposed methodology gives the accuracy score of 64% on both MRPC and SICK dataset 4.2 with minimal pre-processing.

This technique can prove to be very useful in data mining for semantic analysis on text data or clustering of semantically similar documents in order to draw mean-

ingful results from them. This semantic score can used as a feature of documents for performing different analysis on textual data.

In future we can explore different other relationships that exist among the words and phrases. We can exploit parts of speech tags to capture the semantics in more depth. We can also combine one or more relations and project them in a different vector space in order to obtain embeddings such that these embeddings captures the semantics between the words and ultimately the semantics of a document. The score between these embeddings can be computed via min-hashing technique efficiently.

References

[1] S. J. Athenikos and H. Han, "Biomedical question answering: A survey," *Computer methods and programs in biomedicine*, vol. 99, no. 1, pp. 1–24, 2010.

[2] M. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *CoRR*, vol. abs/1508.04025, 2015.

[3] Z. Zhang, Y. Wu, H. Zhao, Z. Li, S. Zhang, X. Zhou, and X. Zhou, "Semantics-aware bert for language understanding," 2020.

[4] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[5] R. Navigli and S. P. Ponzetto, "BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network," *Artificial Intelligence*, vol. 193, pp. 217–250, 2012.

[6] S. J. Nelson, D. Johnston, and B. L. Humphreys, "Relationships in medical subject headings, relationships in the organization of knowledge," *Bean and Green, eds. Kluwer Academic Publishers*, pp. 171–84, 2001.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[8] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[10] A. Z. Broder, "On the resemblance and containment of documents," in *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pp. 21–29, 1997.

[11] F. Ture, T. Elsayed, and J. Lin, "No free lunch: Brute force vs. locality-sensitive hashing for cross-lingual pairwise similarity," pp. 943–952, 01 2011.

[12] K. U. Khan, W. Nawaz, and Y.-K. Lee, "Set-based unified approach for summarization of a multi-attributed graph," *World Wide Web*, vol. 20, no. 3, pp. 543–570, 2017.

[13] M. Manna and G. Abdulameer, "Web documents similarity using k-shingle tokens and minhash technique," *Journal of Engineering and Applied Sciences*, vol. 13, pp. 1499–1505, 05 2018.

[14] R. Rada, H. Mili, E. Bicknell, and M. Blettner, "Development and application of a metric on semantic nets," *IEEE transactions on systems, man, and cybernetics*, vol. 19, no. 1, pp. 17–30, 1989.

[15] R. Richardson, A. Smeaton, and J. Murphy, "Using wordnet as a knowledge base for measuring semantic similarity between words," 1994.

[16] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language," *Journal of artificial intelligence research*, vol. 11, pp. 95–130, 1999.

[17] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *arXiv preprint cmp-lg/9709008*, 1997.

[18] E. G. Petrakis, G. Varelas, A. Hliaoutakis, and P. Raftopoulou, "X-similarity: Computing semantic similarity between concepts from different ontologies.," *Journal of Digital Information Management*, vol. 4, no. 4, 2006.

[19] Z. Zhou, Y. Wang, and J. Gu, "New model of semantic similarity measuring in wordnet," in *2008 3rd International Conference on Intelligent System and Knowledge Engineering*, vol. 1, pp. 256–261, IEEE, 2008.

[20] J. Tang and Y. Tian, "A systematic review on minwise hashing algorithms," *Annals of Data Science*, vol. 3, no. 4, pp. 445–468, 2016.

[21] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan, "On compressing social networks," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 219–228, 2009.

[22] W. Wu, B. Li, L. Chen, and C. Zhang, "Efficient attributed network embedding via recursive randomized hashing," in *IJCAI International Joint Conference on Artificial Intelligence*, 2018.

[23] M. Kuzu, M. S. Islam, and M. Kantarcioglu, "Efficient similarity search over encrypted data," in *2012 IEEE 28th International Conference on Data Engineering*, pp. 1156–1167, IEEE, 2012.

[24] L. Cherkasova, K. Eshghi, C. B. Morrey, J. Tucek, and A. Veitch, "Applying syntactic similarity algorithms for enterprise information management," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1087–1096, 2009.

[25] K. U. Khan, W. Nawaz, and Y.-K. Lee, "Set-based approximate approach for lossless graph summarization," *Computing*, vol. 97, no. 12, pp. 1185–1207, 2015.

[26] W. Wu, B. Li, L. Chen, J. Gao, and C. Zhang, "A review for weighted minhash algorithms," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[27] P. Li and C. König, "B-bit minwise hashing," in *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, (New York, NY, USA), p. 671–680, Association for Computing Machinery, 2010.

[28] A. Shrivastava and P. Li, "Densifying one permutation hashing via rotation for fast near neighbor search," ICML'14, JMLR.org, 2014.

[29] A. Shrivastava, "Simple and efficient weighted minwise hashing," in *Advances in Neural Information Processing Systems*, pp. 1498–1506, 2016.

[30] K. Knight and S. K. Luk, "Building a large-scale knowledge base for machine translation," in *AAAI*, vol. 94, pp. 773–778, 1994.

[31] C. Leacock and M. Chodorow, "Combining local context and wordnet similarity for word sense identification," *WordNet: An electronic lexical database*, vol. 49, no. 2, pp. 265–283, 1998.

[32] P. Chahal, M. Singh, and S. Kumar, "An ontology based approach for finding semantic similarity between web documents," *International Journal of Current Engineering and Technology*, vol. 3, no. 5, pp. 1925–1931, 2013.

[33] M. I. Nasab and R. Javidan, "A new approach for finding semantic similar scientific articles," *Journal of Advanced Computer Science & Technology*, vol. 4, no. 1, p. 53, 2015.

[34] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *arXiv preprint arXiv:1310.4546*, 2013.

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

[36] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext. zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

[37] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[38] M. E. Peters, M. Neumann, R. L. Logan IV, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith, "Knowledge enhanced contextual word representations," *arXiv preprint arXiv:1909.04164*, 2019.

[39] R. Ibrahim, S. Zeebaree, and K. Jacksi, "Survey on semantic similarity based on document clustering," *Adv. Sci. Technol. Eng. Syst. J*, vol. 4, no. 5, pp. 115–122, 2019.

[40] W. B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases," in *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[41] M. Marelli, S. Menini, M. Baroni, L. Bentivogli, R. Bernardi, R. Zamparelli, *et al.*, "A sick cure for the evaluation of compositional distributional semantic models.," in *Lrec*, pp. 216–223, Reykjavik, 2014.

[42] Y. Liu, C.-J. Sun, L. Lin, X. Wang, and Y. Zhao, "Computing semantic text similarity using rich features," in *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pp. 44–52, 2015.

[43] T. Slimani, "Description and evaluation of semantic similarity measures approaches," *arXiv preprint arXiv:1310.8059*, 2013.

[44] Z. Wu and M. Palmer, "Verb semantics and lexical selection," *arXiv preprint cmp-lg/9406033*, 1994.

[45] J. Kamps and M. Marx, "Words with attitude," 01 2002.

[46] R. Rada, H. Mili, E. Bicknell, and M. Blettner, "Development and application of a metric on semantic nets," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 1, pp. 17–30, 1989.

[47] B. Yang and T. Mitchell, "Leveraging knowledge bases in lstms for improving machine reading," *arXiv preprint arXiv:1902.09091*, 2019.

[48] M. E. Peters, M. Neumann, R. L. L. IV, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith, "Knowledge enhanced contextual word representations," *CoRR*, vol. abs/1909.04164, 2019.

[49] A. Z. Broder, "On the resemblance and containment of documents," in *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pp. 21–29, IEEE, 1997.

[50] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[51] Z. Zhang, Y. Wu, H. Zhao, Z. Li, S. Zhang, X. Zhou, and X. Zhou, "Semantics-aware bert for language understanding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 9628–9635, 2020.

[52] A. Broder, "June 1997. on the resemblance and containment of documents," in *Proceedings Compression and Complexity of Sequences*, p. 21.

[53] R. Navigli and S. P. Ponzetto, "Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network," *Artificial intelligence*, vol. 193, pp. 217–250, 2012.

[54] A. Shrivastava and P. Li, "Densifying one permutation hashing via rotation for fast near neighbor search," in *International Conference on Machine Learning*, pp. 557–565, PMLR, 2014.

[55] P. Li and C. König, "b-bit minwise hashing," in *Proceedings of the 19th international conference on World wide web*, pp. 671–680, 2010.

[56] F. Ture, T. Elsayed, and J. Lin, "No free lunch: brute force vs. locality-sensitive hashing for cross-lingual pairwise similarity," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 943–952, 2011.