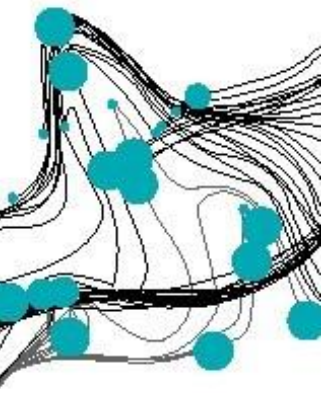
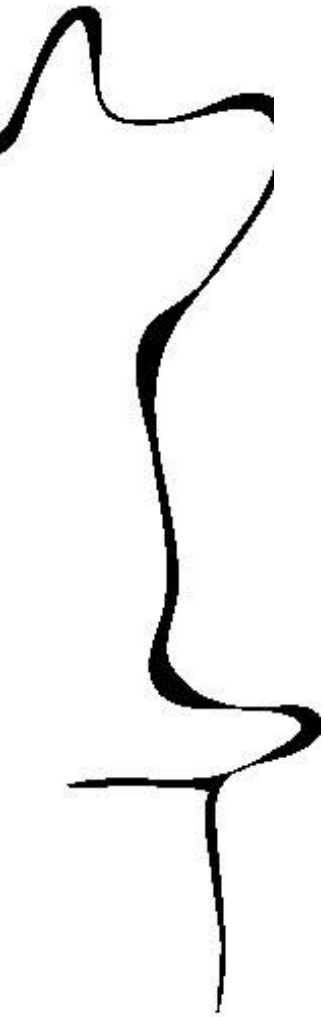


UNIVERSITY OF TWENTE.



SEMANTIC-AWARE EWC CODE RECOMMENDER SYSTEM FOR INDUSTRIAL SYMBIOSIS MARKETPLACE

M.SC. THESIS

Dimas Wibisono Prakoso, s1751425

Graduation committee:
Dr. Chintan Amrit
c.amrit@utwente.nl
Faculty BMS, University of Twente

Dr. Doina Bucur
d.bucur@utwente.nl
Faculty EEMCS, University of Twente

Abstract

The European Union (EU) has established 7th Environment Action program to 2020 as 'living well within the limits of the planet'. To support this program, the EU encourages its members to shift their economic system from a linear economy that focuses on resource use and disposal towards a circular economy. This system encourages maximizing resources by reusing resources within the system. The EU views Industrial Symbiosis (IS) along with eco-design, remanufacturing, and eco-innovation as enabling factors to build the circular economy. IS is defined as a collaboration between company by exchanging materials, energy/utility, water, and by-products as feedstock for an industrial process.

The EU funded a project of web-based IS marketplace platform called Sharebox to stimulate its member in adopting IS. Sharebox users can sell their secondary product or waste by registering it to the system and supplying it with waste item description and appropriate European Waste Catalogue (EWC) code. It is a codification standard that is used by the EU for waste product circulated within the EU region. The code could determine how the product will be handled. A mislabeled code will lead to mistreated of hazardous waste that could harm the personnel and also the environment. The process of labeling a waste item with EWC code is difficult because there are 841 EWC codes which are hard to memorize. Therefore, we need a system that is able to recommend the EWC Code accurately.

This research aims to design methods that can recommend the EWC code accurately for certain waste items. We designed three methods, namely String-based (SB), Knowledge-based (KB) and Corpus-based (CB) EWC Code Recommender System (RS). The SB works by aggregating the string similarity between words contained in the waste item and EWC code description. However, it could not comprehend words and sentences that are lexically different but semantically similar. Therefore, we designed KB and CB methods, which have semantic awareness capabilities to address the problem. KB achieves this by utilizing WordNet-based word similarity, whereas SB by exploiting the relationship between word vectors produced by word2vec algorithm trained on a news corpus.

The experiment result shows the incorporation of semantic-awareness could improve the performance of the EWC Code RS. In Top-10 EWC Code RS, the SB method could achieve recall, precision, and ARHR by 34.4%, 33.9%, and 15.4%. The KB which utilize semantic-awareness could achieve better performance by 38.3%, 35.2%, and 15.4%. The CB perform even higher by 39.2%, 35.9%, and 16.7%. In other words, CB is the best performing method by achieving an increase of 14%, 6%, and 10.4% in recall, precision, and ARHR, respectively. The result is achieved by using general knowledge and corpus resource, which are WordNet and Google News. Both only have decent coverage in dataset since the dataset contain many names and technical terms. We recommend developing an ontology or corpus resource specific to waste or IS field so that it can be used to increase the performance of EWC Code Recommender System

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1. Motivation

The European Union (EU) has established a vision of 2050 as 'living well within the limits of the planet' in the 7th Environment Action program. To achieve this vision, the EU encourages its members to shift their economic system from a linear economy that focuses on resource use and disposal towards a circular economy. This system encourages maximizing resources by reusing resources that are in the system. The EU views Industrial Symbiosis (IS) along with eco-design, remanufacturing, and eco-innovation as enabling factors to build the circular economy [1].

IS is defined as a collaboration by exchanging materials, energy/utilities, water, and by-products from one company as feedstock for an industrial process in another company [2]. The collaboration gives economic and environmental benefit to the involving parties. In 2011, COWI estimated the market potential of Industrial Symbiosis in Europe by extrapolating the National Industry Programme. They estimated that an investment of EUR 250 million (as operating costs of the program) would generate savings of EUR 1,400 million as well as environmental benefits of 52 million tons of landfill diversion and 45.5 million tons of CO_2 reduction [3]. Considering the benefit, IS has been studied by numerous research discipline including economy, engineering, material exchange, social, organizational theory, and information system.

There are many approaches that can stimulate industrial symbiosis to emerge in the industrial community, which is coordinating bodies, self-organizing, and facilitated approaches [4]. In coordinating bodies, the authorized entity such as local governments will connect companies in their region that are identified as having the potential to make waste trade. In regions where authorities lack initiative, companies can self-organizing waste trade if they identify that there are business benefits which can be obtained by doing so. In the latter way, an expert intermediary is needed to identify the IS potential and then connect and facilitate waste trade within or between companies.

In relation to the facilitated approach, the information system can be used as tools to facilitate IS identification [5]. There are five types of information system for IS identification, which is open online waste markets, facilitated synergy identification system, industry sector synergy identification, social network, knowledge repositories, region identification. The open online waste market is a web-based platform where users can engage in business-to-business waste trade. The EU utilize this kind of information system to stimulate the development of IS. The EU funded a project called Sharebox[1] , which is a web-based platform where plant operator and product manager can monitor and trade their by-products with their supplier or with other companies in industrial symbiosis manner . Sharebox will be used as a study case of this research.

The initial phase for waste trading in Sharebox is registering the waste product with description and label it with the appropriate European Waste Catalogue (EWC) code. The waste code labeling is beneficial to reveal IS opportunities. If the waste product (output of industry) and ingredients of an industrial process (input) have been labeled by EWC code, input-output matching can be executed easily by matching the codes. However, the task of labeling the waste product with the correct EWC code is hard and time-consuming. The EWC standard has hundreds of code entries which make it hard to memorize and to browse manually by the user. To make the task easier, we develop the EWC code recommender system (EWC RS). The EWC RS is a system that. can recommend EWC code to the user who inputs waste product description. Our focus of this research will be on building such a system.

[1] http://sharebox-project.eu

The contribution of our research to the field of industrial symbiosis are three folds. First, we design methods that are able to address a problem of how to accurately recommend EWC codes. The method comprises pre-processing step using Natural Language Processing (NLP) such as tokenization, stemming, and lemmatization and recommendation generation step that exploit WordNet-based word similarity and word embedding. The method then can be used further in an online IS marketplace platform such as Sharebox system. Second, we can determine to what extent adding semantic awareness could improve the performance of EWC codes recommender. Semantic awareness is an ability that can comprehend semantically similar short text in the process of generating a recommendation. Third, we can also determine how the general lexical ontology such as WordNet and news corpus could improve the performance of the recommender in IS field. To the best of our knowledge, ontology, or corpus that are built specifically for industrial symbiosis, waste or environmental field does not exist. Additionally, our research also contributes by providing a systematic literature review (SLR) to understand the state of the art of methods for determining short-text similarity (STS). The SLR contains a brief description of the techniques, including strength and weaknesses. It can be used as a reference to select the appropriate STS methods to solve a certain problem or to devise a new method. We conduct this SLR as a preliminary process to design the EWC code recommendation since the core of the methods itself is a short text similarity comparison between waste item and EWC code description.

1.2. Problem definition

In Sharebox, if the user wants to sell the waste product, the user must register the product in the system by inputting the name of the company producing the waste, description of waste products in the form of free text and also labeling the product with the proper EWC code. This code is taken from a catalog containing a list of hundreds of EWC code entries where each code has its own description. Waste products need to be labeled with the EWC code, which code description is considered relevant with the description of the waste product. Manual labeling will be difficult because there are many EWC code entries that the user must remember or browse. Users require a system that is able to recommend the relevant EWC code. Therefore, this research tries to solve a problem, which is how to accurately recommend the relevant EWC code when given waste product description on the IS open online marketplace.

From the problems described above, we formulate the **main research questions (RQ)** as follows.

Given a waste product description in IS marketplace, can we accurately recommend EWC code that the product belongs to?

We divided the main RQ into several sub-questions (SQ) so that the research will be more focused, and the main RQ can be answered appropriately.

SQ1: What recommender system method is suitable with the conditions where the user interest is difficult to obtain due to the limited information of user-item interaction?
In the context of the recommender system, our dataset contains only a few users and a limited history of interaction between the user and the item (EWC code in this case) that is selected. There is not enough information to extract user interest in items. General personalized recommendation systems such as Content-based (CB) and Collaborative Filtering (CF) require this user interest / profile to provide recommendations. In CB, items similar to user interest will be recommended while in CF, items that are liked by other users who have an interest similar to current user interest will be recommended. Even though there is no adequate transaction history, the user interest can still be

extracted from the description of the waste product. The challenge is how to determine the type of recommendation system that is suitable for this situation.

SQ 2: What short text similarity measurement method that is available in the literature?
In recommending related EWC code, there is a challenge on how to determine the EWC code description that is relevant to the waste product description. A method for comparing the relevance or similarity between those two short text needs to be researched in the literature.

SQ3: What is the effect of incorporation of semantic-aware short text similarity measurement method to the accuracy of EWC RS?
Short text similarity (STS) can be measured not only in lexical / string similarity but also in semantic meaning. EWC code 160117 has a description of "Ferrous metal". This code must be recommended to the waste product that has a description of "Iron and steel scrap". Even though "Iron and steel scrap" and "Ferrous metal code" are lexically different, they have a semantically similar meaning. This research will investigate the effect of using semantic-aware STS measurement methods to the accuracy of EWC RS.

SQ4: How does short text preprocessing (e.g., stemming, lemmatization) affect the accuracy of EWC RS?
Stemming and lemmatization is normalization of a word to retrieve its basic form. The difference is that stemmer only reduces the inflection while lemmatization also considers word context and look-up dictionary to derive the word basic form while. For example, for the word saw, stemming might return s while lemmatization could return see (as a verb) or saw (as a noun) depending on the word context in the sentence. This research will incorporate stemming, and lemmatization in preprocessing step for sentence similarity measurement then investigate its effect on the accuracy of EWC Code RS.

SQ5: How does the word similarity method affect the quality of EWC RS?

The core of EWC Code RS is the comparison between the description of the waste item and description of EWC Code. The description text comprises words. This research also will try to reveal what word similarity method that gives the best performance of the recommender

1.3. Report organization

The remaining thesis is organized as follows. Chapter 2 discuss the methodology used to conduct the research. Chapter 3 discuss the overview of related work of recommender system and short text similarity measurement method. This chapter provides answers to SQ1 and SQ2. Chapter 4 explain the experimental setup, including dataset, model, data preprocessing, and evaluation method. To answer SQ3, SQ4, and SQ5, we provide Chapter 5 that contain experiment result. The result is discussed in Chapter 6, and the conclusion is drawn in Chapter 7.

2. Methodology

2.1. Design Cycle

We view our research as a design of a method to solve a problem. Therefore, we apply the Design Cycle method, which is a part of Design Science Research methodology introduced by Wieringa [6]. Design Cycle comprises three steps, which are problem investigation, treatment design, and treatment validation. If we add the treatment implementation step to the cycle, it will form the engineering cycle as illustrated in Figure 1. Treatment implementation itself means to transfer the

method to the real-world context, which we will not cover in this research. By adapting the Design Cycle, our research methodology can be explained in more detail in the following sections.

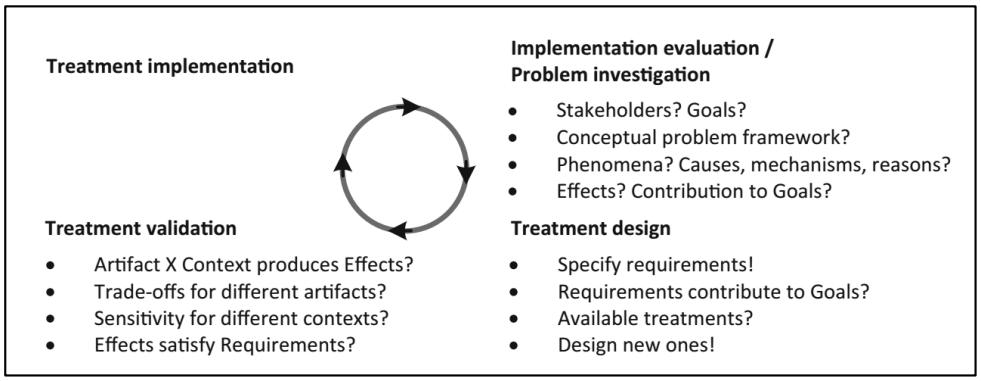


Figure 1. Engineering Cycle

2.2. Problem Investigation

In this step, we formulate the problem that has to be solved and what goal to be achieved. The current situation must be investigated so the appropriate solution can be made. The problem needs to be solved defined in the Problem Definition and Research Question section.

We also conduct analysis on our datasets to narrow down the possible solution that might be fit with the characteristic of our dataset. Our data comprises two datasets. The first dataset is IS dataset that contains waste product input by the user while the second dataset is EWC dataset that contains EWC code and its description. The datasets are in the form of short text with a maximum length of 20 characters. Most of it is not a complete sentence that contains Subject and Predicate but just a Noun phrase. Some of the waste items in IS dataset has been labeled with an EWC code as historical data. The datasets will be further explained in section 4.2 (Dataset).

From the explanation about the datasets above and Problem Definition and Research Question Section, we need to find a solution on how to measure short text similarities and recommend items. Therefore, we try to find the solution from the literature or devise a new one if it is more appropriate. We conduct a literature study on the Recommendation System to get a better understanding of how it works. We also conduct a literature study on Natural Language Processing (NLP) since it offers a technique that can be applied in our method design such as stemming, lemmatization, edge, and node-based word similarity. The work in NLP area also has invented techniques to measure short text similarity. We conduct a Systematic Literature Review (SLR) on these methods to grasp a holistic view of the field. Literature study step will be explained in more detail in Section 3.

2.3. Treatment Design

In this step, we develop the method as the artifact. The artifact is the method that can recommend waste code (EWC standard) to be selected by the user of the online open waste marketplace platform.

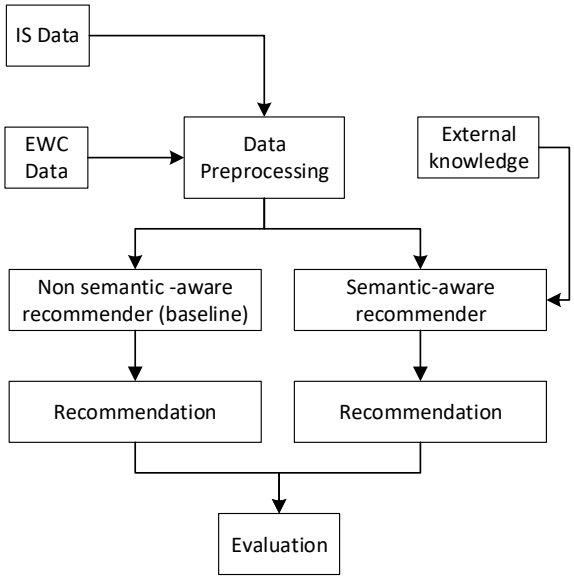


Figure 2. A general view of the model describing the recommendation approach

The method in Figure 2 can be explained as follows. As an information source, there are two types of datasets. The first is IS data that contain user ID (company), the user waste description, and EWC code. A fraction of this data is already labeled with EWC code by the user. This labeled data will be used as test data. Then, data processing is conducted by using NLP techniques such as tokenization, stop word removal, and stemming. The dataset is used as input for a Non-semantic-aware RS, which is developed by comparing waste item description with EWC code description. The comparison will utilize STS measurement method based on string similarity to be able to capture the semantic relation between words that are lexically different. This RS will be used as a baseline. A semantic-aware RS that can capture semantic meaning will be developed. It achieved this capability by exploiting external knowledge such as lexical ontology (WordNet) or Google News. The method to implement such a technique will be researched from the literature. For both type of RS, if the similarity value between waste product description and EWC code description exceeding certain thresh hold, then the current EWC code is returned back as a recommendation. Section 4 explains the process in more detail.

2.4. Treatment validation

The proposed method is instantiated in Python programming language. We choose Python because many libraries to develop NLP task and Recommender system are available in Python. We measure the performance of the Recommender Systems by using the offline evaluation metric such as recall, precision, and average reciprocal hit rank (ARHR). More detailed evaluation method can be seen in section 4.6 (Evaluation Method).

3. Literature Study

3.1. Recommender System

In today's era of abundant information, users easily experience information overload. The recommendation system emerges to overcome this problem. The recommendation system can be defined as a system that can recommend the most relevant item for a particular user by predicting user interest in items by utilizing information about items attributes, users information, and history of users-items interactions [7]. Recommendation systems can take various forms depending on the case and the domain of the problem. The most common types are Collaborative Filtering, Content-based, and Hybrid system.

3.1.1. Collaborative Filtering

Collaborative Filtering (CF) is a popular recommendation technique which prediction and recommendation for the active user (to whom the technique tries to recommend item) are based on an aggregation of active user's and or other users' interest toward items obtained from the history of user-item interaction [8]. There are two types of CF approach exist in the literature, namely user-based and item-based. The former is introduced firstly by Grouplens in 1994 [9] while the latter is proposed by Amazon in 2003 [10].

3.1.1.1. User-based Collaborative filtering

In user-based CF, interest from active users is determined by other users who have the same taste or the same rating pattern. Items that are liked by these users are most likely to be liked by active users as well. The extent of active user interest to a particular item is determined by aggregating similar user's interest towards the item.

An illustration of how user-based CF works is given in Figure 3. According to the history of user-item interaction, there are interactions between three users and five items. An arrow pointed from user to items indicates that the user liked the item. The system tries to recommend items to user 3 as an active user. The figure shows that user 1 liked item A, B, and C. User 2 liked the different item, which is C only. User 3 is the active user who has liked item B, and C. User-based CF assumes that similar users will also share items they like. User 1 and user 3 like items in common which is item B and C. From this fact, it can be concluded user 1 are highly correlated or similar with user 3 because they have similar rating pattern. If user 1 like item A, then user 3 will be most likely interested in item A as well. Therefore, item A will be recommended to user 3.

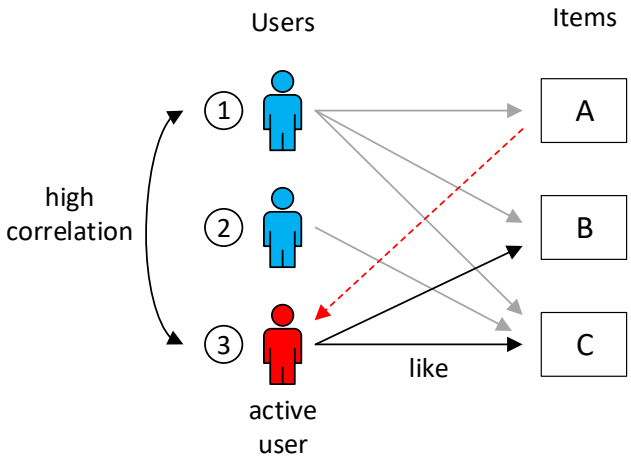


Figure 3. The principle of a user-based collaborative filtering recommender system

In a more detailed process, user-based CF comprises several steps as follows. Firstly, the techniques will try to find similar users with the active user by using a metric of similarity, such as the Pearson correlation coefficient. Assume S_{xy} is a set of items that are liked by both user x and y. $r_{x,s}$ and $r_{y,s}$ are assigned a rating of both users on item s. $\bar{r}_x$ and $\bar{r}_y$ are average rating to all items by user x and user y. Then similarity of user x and y or $sim(x, y)$ can be calculated using Pearson correlation by the equation (1). Another alternative to calculating similarity is by using (raw) cosine similarity. Each user is represented as a vector with his rating as its element. Then, the cosine angle between the two vectors is calculated using equation (2). The smaller the cosine angle, the more similar the users are. An extension to this approach is adjusted cosine where

user's rating average to all items is also taken into account as defined in equation (3). μ_x and μ_y denote average rating of user x and user y for co-rated items of both users.

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)(r_{y,s} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)^2 \sum_{s \in S_{xy}} (r_{y,s} - \bar{r}_y)^2}} \quad (1)$$

$$sim(x,y) = \cos(\vec{x},\vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\|_2 \times \|\vec{y}\|_2} = \frac{\sum_{s \in S_{xy}} r_{x,s} r_{y,s}}{\sqrt{\sum_{s \in S_{xy}} r_{x,s}^2} \sqrt{\sum_{s \in S_{xy}} r_{y,s}^2}} \quad (2)$$

$$sim(x,y) = \cos(\vec{x},\vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\|_2 \times \|\vec{y}\|_2} = \frac{\sum_{s \in S_{xy}} (r_{x,s} - \mu_x)(r_{y,s} - \mu_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \mu_x)^2} \sqrt{\sum_{s \in S_{xy}} (r_{y,s} - \mu_y)^2}} \quad (3)$$

Secondly, after all similar users are obtained, the method predicts active user interest or rating to unrated items by aggregating rating of similar users to the items. [11] describes several common methods to calculate the active user's rating as defined in equation (4), (5), and (6). $r_{c,s}$ denotes rating of user *c* assigned to item *s*. *N* is the number of similar users. $\hat{C}$ is set of similar users and $r_{\acute{c},s}$ is the rating of a similar user to item *s*. *k* is a normalizing factor and is defined by $k = \frac{1}{N \sum_{\acute{c} \in \hat{C}} |sim(c,\acute{c}|}$. $\bar{r}_{\acute{c}}$ is the average rating of a similar user to all items. Finally, after all of the active user rating to the unrated items have been determined, the top N rated items are chosen as recommended items.

$$r_{c,s} = \frac{1}{N} \sum_{\acute{c} \in \hat{C}} r_{\acute{c},s} \quad (4)$$

$$r_{c,s} = k \sum_{\acute{c} \in \hat{C}} sim(c,\acute{c}) \times r_{\acute{c},s} \quad (5)$$

$$r_{c,s} = \bar{r}_c + k \sum_{\acute{c} \in \hat{C}} sim(c,\acute{c}) \times (r_{\acute{c},s} - \bar{r}_{\acute{c}}) \quad (6)$$

To illustrate the process, consider Figure 4 as an example case. There are four users who like five items. Their interest in the items is represented by an interval-based rating from 0 to 5. The increasing value of rating means an increasing level of user interest toward the item. The empty cells mean the user has not been rated the items. We also set user 1 as the active user. The rating of his unrated items such as item C will be predicted by the algorithm. The predicted rating will determine whether the item will be recommended or not.

		Item				
		A	**B**	**C**	**D**	**E**
User	**1**	4	2		5	3
	2	5	3	5		4
	3	3		3	5	2
	4		5	1	1	2

Figure 4. user-item matrix example

The first step in user-based CF method is that it will try to identify the most similar users with user 1. If Pearson's correlation coefficient is used in this case, the similarity between user 1 and user 2 is calculated using equation (1):

$$sim(1,2) = \frac{(4-3.5)(5-4.25)+(2-3.5)(3-4.25)+(3-3.5)(4-4.25)}{\sqrt{((4-3.5)^2+(2-3.5)^2+(3-3.5)^2)((5-4.25)^2+(3-4.25)^2+(4-4.25)^2)}} = 0.97$$

With the same equation, similarity among users can be seen in Figure 5. Figure 6 shows the user similarity matrix using adjusted cosine similarity. From both figures which use different similarity equation, we can conclude that the most similar user with user 1 is user 2 and user 3.

User	1	2	3	4
1	1.00	0.97	0.87	-0.89
2	0.97	1.00	-0.04	-0.96
3	0.87	-0.04	1.00	-0.40
4	-0.89	-0.96	-0.40	1.00

Figure 5. User similarity matrix using Pearson's correlation coefficient

User	1	2	3	4
1	1.00	1.00	0.98	-0.89
2	1.00	1.00	1.00	-0.96
3	0.98	1.00	1.00	-0.76
4	-0.89	-0.96	-0.76	1.00

Figure 6. User similarity matrix using adjusted cosine similarity.

After all similar user with the active user has been identified, the second step in the CF method is the prediction of the unrated item. Predicted rating of item C for user 1 is calculated using equation (4) is as follows. Calculation using equation (5) and (6) are also provided as comparison purpose. From the calculation using that three formulas, the predicted rating of user 1 for item C are 4, 4.03. and 3.78 which can be rounded to 4. A similar calculation is applied if there are still any unrated items by user 1.

$$r_{1,C} = \frac{1}{2}(5+3) = 4$$

$$r_{1,C} = 0.54((0.97 \times 5) + (0.87 \times 3)) = 4.03$$

$$r_{1,c} = 3.5 + 0.54((0.97 \times (5-4.25)) + (0.87 \times (3-3.25))) = 3.78$$

The third or final step in user-based CF is select the N top predicted rated for the active user. Since there is only one item with a high predicted rating, which is item C, then item C is selected as a recommended item.

3.1.1.2. Item-based Collaborative Filtering

Item-based CF basic principle is that interest of the active user to an item is determined by the aggregation of his interest towards similar items. To illustrate the concept, consider Figure 7. There are three users and three items with arrows that represent the user-item to items. User 3 is set as an active user, and he has liked item C in from user-item interaction history. By using similarity metrics, it is known that item C has a high similarity with item A, so it is highly correlated. If item C is liked by user 3, then it will be most likely that item A will be liked by user 3 as well. Therefore, item A will be recommended to user 3.

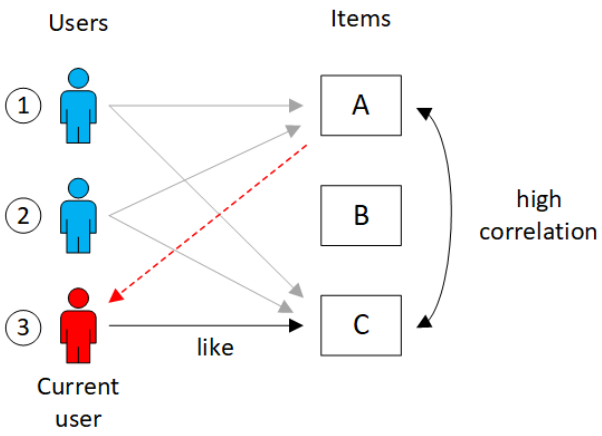


Figure 7. The principle of item-based collaborative filtering recommender system

Detailed steps of how item-based CF works is similar to user-based CF but with a changing perspective from user similarity to items similarity. Firstly, item-based CF will try to identify similar items with the item which rating is tried to be predicted. The similarity metrics in user-based CF (e.g., Pearson's correlation, cosine, adjusted cosine) are also applicable in this case. Secondly, the rating to the unrated item is calculated by aggregating rating pattern from active user towards similar items. And finally, items with the Top-N predicted rating is set as recommended items.

3.1.2. Content-based

The Content-based Recommendation (CB) technique recommend items which are similar to items previously liked by a user. The methods can be illustrated in Figure 8.

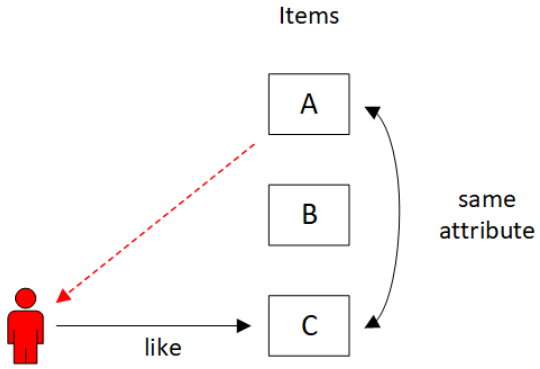


Figure 8. The principle of a content-based recommender system

Basic principles from the Content-based recommendation system are: 1) Analyzing the item description preferred by certain users to determine the common principal attributes (preferences) that can be used to distinguish these items. This preference is stored in the user's profile. 2) Compare the attributes of each item with the user profile so that only items which have a high level of similarity with user profiles will be recommended [12]. As an example, in Figure 8, we can see that user A liked item C in the past. The recommender system will search item that has a similar attribute such as item description. The system found that item A has a high degree of similarity with item C; therefore, the item is returned as a recommendation for the active user.

This Content-based technique has the advantage of being able to analyze products and find similarities with a product that the active users liked in the past to recommend the item. Unlike CF, this technique does not require an extensive list of other users' item selection history [10]. However, sometimes, sophisticated techniques are required to analyze the content of complex items such as audio and video.

3.1.3. Hybrid

To overcome the weaknesses of each method, CF and CB can be combined into a hybrid system. According to Burke [13], hybridization methods can be classified into seven categories, which are:

- Weighted: Add scores from different recommender components.
- Switching: Choose methods by switching in different recommender components.
- Mixed: Show recommendation result from different systems.
- Features Combination: Extract features from different sources and combine them as a single input.
- Feature Augmentation: Calculate features by one recommender and put the result to the next step.
- Cascade: Generate a rough result by a recommender technique and recommend on the top of the previous result.
- Meta-level: Use the model generated by one recommender as the input of another recommender technique.

Even though combining methods can yield better recommender theoretically, there might be other factors specific to domain problem that must be considered.

3.1.4. Knowledge-based

Both CF and CB recommender system requires user history of a past selection of items. In the CF method, even a higher number of interactions between users and items is needed to cover a wider spectrum of items to be recommended. During the initial system deployment or because of the characteristics of the system, sometimes this is not available. This is known as a cold-start problem. Knowledge-based RS emerge to overcome this problem. This system is considered a special case of CB, where it still generates recommended item based on item attributes. But instead of matching the item attributes with a history of past interaction between user and items (user ratings), it utilizes user requirement/specification for items at a certain moment[14]. User requirement is explicitly stated by the user through the interface to the system. The difference can be summarized in Table 1.

Table 1. The conceptual difference of recommendation system approaches

Approach	Conceptual Goal	Input
Collaborative	The recommendation is given based on a collaboration of interest of active users and other users	User ratings + community rating
Content-based	The recommendation is given based on the interest of the active user and content of the items he liked.	User ratings + Item attributes
Knowledge-based	The recommendation is given based on the interest of the active user given by user specification at a time (domain knowledge)	User specification + Item attributes + domain knowledge

	not the history of the item he liked in the past.	

In our problem domain, the interaction history between the user and the EWC code is very limited. Each user only chooses one or two EWC codes. This causes the rating of users and communities to be very difficult to determine. Users interest can also change at any time depending on the description of the waste item entered, regardless of the EWC code that was previously chosen. These characteristics make the problem domain unsuitable to be solved by a collaborative and content-based approach. Knowledge-based is more suitable because the description of waste items can be derived to obtain user specifications. The compatibility between this user specification and the attribute item (EWC code description in our case) will be used to provide EWC code recommendations.

3.1.5. Semantic-Aware Recommender System

In the context of a recommender system, researchers have been proposed numerous techniques to incorporate semantic awareness into a recommender system. de Gemmis et al. [15] classify the approaches to apply semantic capability to CB recommender system into two main types, which are Top-down and Bottom-up [15]. The top-down approach utilizes external ontology to capture the semantic meaning of item content. An external ontology that can be used for example is WordNet (for linguistic) or Wikipedia. The Bottom-up approach works with the principle that terms or are closely related if they are located in the same context or space. The techniques that are commonly used are LSI, Word2Vec using large corpora.

The approach described above can also be beneficial for our domain problem. In the process of producing a recommended EWC code, there is a necessity to incorporate semantic awareness capability. As an illustration, consider waste item description *iron and metal waste*. The user labels this waste item with EWC code description *ferrous metal* (EWC code: 16 01 17). Without semantic capability, the EWC code will not be recommended because there no shared term between those two descriptions while it is obvious that both descriptions are semantically related. By adding semantic awareness to EWC recommender system, the performance of the system can be expected to increase.

3.2. Systematic Literature Review on Short-text Similarity Methods

In our EWC code recommender system, the waste item description will be compared with the EWC code description, and the similarity will be measured. Codes with the most similar descriptions will be returned by the system as recommendations. Based on that requirement, there is a necessity to apply techniques to measure the similarity between short text. Therefore, we conducted a Systematic Literature Review (SLR) to find out what techniques are available in the literature, including characteristics, weaknesses, and shortcomings. By knowing this, we can choose suitable techniques for our problems.

3.2.1. SLR Method

We follow SLR guideline provided by Kitchenham et al. [16], which is de facto standard for literature review in the software engineering field. The guideline mainly comprises three phases, which are Planning, Conducting, and Reporting. In the Planning phase, a review protocol is defined. It specifies the methods that will be used to undertake a specific systematic review. The protocol comprises the definition of rational of the survey, research questions, search strategy,

study selection criteria and procedure, study quality assessment, data extraction strategy, and data synthesis. After a review protocol is defined, conducting phase are executed by following that protocol. We combine the guideline with the snowballing approach based on guidance by Wohlin [17] . After a primary study is defined, we conduct forward and backward snowball to expand the coverage of the literature search. The expansion might find literature that also relevant to the research questions.

3.2.2. Search strategy and resource database

Having defined the research questions in the previous section, we designed a search string based on our research questions. We also use alternatives and synonyms for each term and linked them all by the use of AND/OR Boolean expressions to cover more search results. The following search string is used to find relevant studies in the paper's title, keywords, and abstract.

("short text" OR text OR sentence) AND similarity AND (method OR algorithm OR measure) AND (syntactic OR lexical OR semantic) AND (corpus OR semantic net OR knowledge)

After search terms are constructed, we conduct a primary search by using the search terms to databases that we consider as the main resource for the computer science field. The database that we used and the search result are summarized in Table 2. We found 3,398 potential primary studies.

Table 2. Database and Search result

Database	Search result
IEEEXplore (http://ieeexplore.ieee.org)	374
ACM Digital Library (http://dl.acm.org)	620
Springer Link (http://www.springerlink.com)	1,747
Science Direct (https://www.sciencedirect.com)	657
Total	3,398

3.2.3. Study selection

Based on the search results, we performed the secondary search by evaluating the studies (identified by primary search) based on their titles, abstracts, and conclusions. Then we used the following inclusion and exclusion criteria to select the relevant primary studies.

Inclusion criteria:

1. The study is peer-reviewed.
2. The study is about a technique that can be applied for short text.
3. It is relevant to the search terms defined in Section 3.1
4. The study includes a detailed empirical evaluation.
5. If more than one paper reports the same study, only the latest or fullest paper was included

Exclusion criteria:

1. Abstract papers with no full-text available are excluded.
2. The study is reported in the non-English language.
3. Short papers with less than four pages are excluded.
4. Duplicated studies (by title or content)

At the end of the study selection process where primary studies have been identified, we applied forward and backward snowballing method by Wohlin [17] to extend the coverage of the search result. The overall selection phases are summarized in Figure 9.

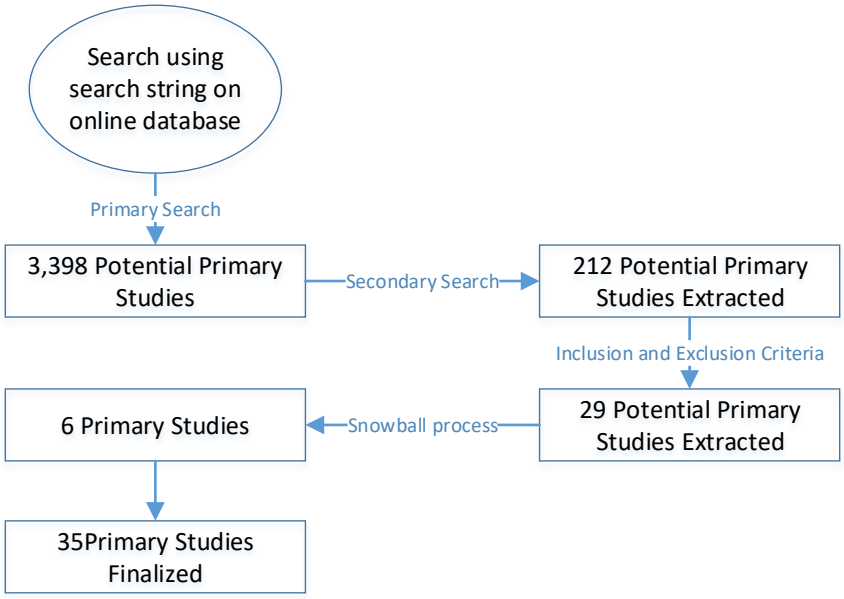


Figure 9. Study selection process

Primary search using string search produced 3,398 studies. The number of studies was then significantly reduced in the secondary search stage, which examined the title, abstract, and conclusion. Then we applied inclusion and exclusion criteria so that the potential primary study was reduced further to 29 papers. Backward and forward snowballing were applied to references, resulting in 6 additional studies. In total, the study selection process produced 35 primary studies.

3.2.4. Study quality assessment

Additionally, in the process of study selection, we also specified the following quality assessment criteria so that the SLR could produce reliable and high-quality result and conclusion.

- Criteria 1: Study contribution is clearly described.
- Criteria 2: Artefacts and methods used in the study are clearly described.
- Criteria 3: Empirical validation is performed.
- Criteria 4: The results and applications are described and discussed thoroughly.

3.2.5. Data extraction and synthesis

After 35 primary studies were obtained, we extracted relevant data from the papers to answer the research question. Additionally, we also extracted data to compile bibliographic information. The types of data we extract from our paper are summarized in Table 3.

Table 3. Data extracted from the paper

Type of the data	Description
Study ID	Unique ID for each paper
Year	The year when the paper was published
Author	The author of the paper
Title	The title of the paper

Venue	Publication venue of the research, e.g., conference proceeding, journal
Technique	Characteristics and techniques used by STS measurement methods
Semantic knowledge and corpus used	Semantic knowledge or corpus utilized by STS measurement methods
Strengths and weaknesses	STS methods capability, determined from aspects such as domain and language independence, the requirement of semantic knowledge, corpus and training data and capability to identify semantic meaning, word order similarity and polysemy
Result	Dataset, experiment setup and result to assess the STS methods performance

In term of publication time, Figure 10 shows the distribution of 35 primary studies per year.

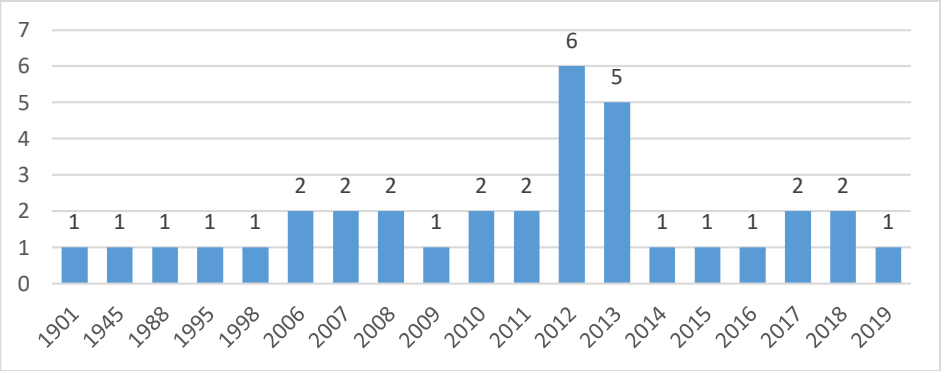


Figure 10. Distribution of primary studies per year

We could see several papers were published before 2006. The papers were about the classic method of STS measurement, which only compares sequences of characters or words without taking into account the semantic meaning of the sentence. For the following years, the publication of papers in this field was relatively stable except in 2012 and 2013. On that year, there was a significant increase due to the existence of the SemEval 2012 conference. At this conference, there was one competition named Semantic Text Similarity, where 88 methods were submitted [18]. However, for this SLR, we only reviewed methods that were ranked in the top 3

3.2.6. SLR Result

3.2.6.1. String-based methods

STS methods that fall into this category measure sentence similarity based solely on character or string sequence that built up the sentences. It does not rely on external semantic net or corpus to do the similarity calculation.

Sentence similarity can be measured by calculating the longest common substring shared by both sentences in comparison. The higher the degree of Longest Common Substring, the more similar the sentences are. Ukkonen [19] proposes an algorithm to calculate the Longest Common Substring by using a generalized suffix tree. Another extension of Longest Common Substring is Longest Common Subsequence. The difference is that in the previous concept, the character sequence must be a combination of adjacent characters while in the latter concept, the characters may not be adjacent, but the order must be the same. Elhadi [20]

introduces a method to calculate text similarity by comparing the Longest Common Sequence between two texts.

Sultana and Biskri [21] propose another method by utilizing the N-grams of characters. N-grams are a subsequence of characters or words that are contained in a sentence or text. First, the method chunks the two sentences being compared into a combination of n-grams of characters with all the possible size of n (the maximum result can be achieved by trigram from the experiment). Then it puts the n-grams into a distance matrix for each sentence. A cell in the matrix contains the distance from an n-gram to another n-gram within a sentence. Finally, sentence similarity is measured using the Jaccard coefficient [22] between those two distance matrix. The method is tested in a sentence comparison task following the experiment set up in [23]. It achieves an accuracy of 89,796%. The advantage of this method is that it can be used for any language and domains since it does not rely on semantic ontology or corpus collection. Even though it yields an encouraging result, this method possesses limitation such as it cannot detect passive sentence and semantically similar sentence.

Sentence similarity can also be measured by comparing common terms that are shared by both sentences. Jaccard coefficient calculates similarity by counting the number of shared terms and divide the count by the number of joint terms of both sentences [24]. A similar approach is used by the Dice coefficient, but it uses a different calculation. The similarity is computed by counting the number of common words, multiply it by two and divided by a total number of terms in both sentence [24].

Salton et al. [25] introduce a vector space model that can be used for sentence similarity measurement. Sentences are transformed into sentence vectors in the vector space model, as illustrated in Figure 11.

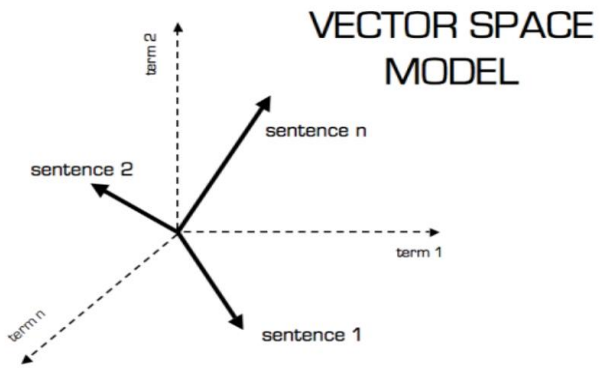


Figure 11. Vector Space Model

The element of the vector is the terms/words that compose the sentences. Formally, if we want to measure the similarity of sentence D and sentence Q, both sentence can be written as

- $D = (t_0, w_{d_0}; t_1, w_{d_1}; \dots; t_t, w_{d_t})$
- $Q = (q_0, w_{q_0}; q_1, w_{q_1}; \dots; q_t, w_{q_t})$

t_k represent term and w_{d_k} or w_{q_k} denotes the weight associated with the term that provides the degree of importance of that term for sentences representation. w_{d_k} is computed using

the Term Frequency-Inverse Document Matrix (TF-IDF) scheme from [26]. To measure the sentence similarity, Salton et al. use cosine vector similarity by using equation (7)

$$similarity(Q, D) = \frac{\sum_{k=1}^{t} w_{q_k} \cdot w_{d_k}}{\sqrt{\sum_{k=1}^{t} {w_{q_k}}^2} \sqrt{\sum_{k=1}^{t} {w_{d_k}}^2}} \quad (7)$$

3.2.6.2. Knowledge-based methods

Knowledge-based methods utilize a network of concepts/terms that are semantically interrelated to extract similarity between words before scaling up into sentence level. The semantic network is varied and can be specific to certain domains such as biomedicine, law. However, if it is not available, general-purpose semantic networks such as WordNet can be used. WordNet is a lexical ontology that is similar to a dictionary that contains the concepts or words, and its definition [27]. Every concept or word that has the same meaning is grouped in synonym set or synset. Each synset is connected in a relationship that forms a semantic network/taxonomy. The relationships can be in the form of *a-part-of*, *a-kind-of*, *is-the-opposite-of*. We can find numerous formula to measure the degree of relatedness between the concept in the semantic network including Path algorithm [28], Leacock and Chodorow (LCH) [29], Wu and Palmer (WP) [30], Resnik [31], Lin [32] and Jiang and Conrath (JCN) similarity [33].

To scale up to sentence level, we need methods that can utilize concept similarity measurement above. Liu and Wang [34], Croft [35], and Li [36] use a similar approach to measure sentence similarity. First, they create a joint word set from both sentences. Second, they generate sentence or semantic vectors by using the joint word set as vector vocabulary and finally measure the similarity by calculating the cosine coefficient between the sentence vectors. However, the difference lies in the second step. Liu and Wang generate each component of the semantic vector by calculating the maximum similarity value of word pair between every word in the joint word set and every word in a sentence. To measure word-pair similarity, they develop their similarity measures based on concept vector. In Croft [35], a sentence vector component is created by summarizing word-to-word similarity value between the corresponding word with a term in joint word. It also exploits the word-to-word similarity from Rada et al. [28]. Li et al. [36] use Lin algorithm [32] as word similarity metric and consider verb and noun type in their sentence similarity calculation. To measure the overall sentence similarity, they combine semantic and word order similarity.

A different approach is taken by Castillo and Cardenas [37]. They tokenize the sentences being compared into two lists of a token. Word by word similarity from both token lists is measured using word similarity from Resnik [31], Lin [32], Jiang and Conrath [33], and Pirro and Secco [38]. Then, the problem of similarity between two lists of words is transformed into a bipartite graph matching and solved by using the Hungarian algorithm [39]. Finally, sentence similarity is measured by summing optimal assignment in the graph divided by the maximum number of the token between the two lists of the token.

Wang and Taylor [40] using a technique called concept forest as a basis for text similarity. The method starts by extracting keywords from both texts being compared and stem the keywords into the base form of the word without inflection. In each document, each keyword is compared to each other semantically by utilizing WordNet. All terms that can be related in WordNet is grouped and forming a tree-like hierarchical structure called concept forest. The

text similarity is measured by comparing concept forest from both texts using the Jaccard index.

3.2.6.3. Corpus-based methods

Corpus-based methods use an external corpus to extract the relation between words or text. Some methods derive the relation between words from a large corpus and then aggregate this relation to measuring similarity in higher extend or sentence level. While the other methods can measure text similarity directly without the process of scaling up.

O'Shea et al. [41] applied Latent Semantic Analysis (LSA) [42] to measure text coherence. Initially, it is intended for a large document, but it is also applicable for short text or sentence. LSA assumes that related words will co-occur in the same context/paragraph. LSA derives the relation between words and context from a large collection of a corpus and represents this relation in the form of the word by context matrix. An entry in the matrix means that a word is present in a particular context. The resulting matrix could be in very high dimension, which is very computationally expensive. Thus, the matrix dimension needs to be reduced. The method decomposes the matrix using singular values decomposition (SVD) into three others matrices, including a diagonal matrix of singular values. This diagonal matrix is truncated by deleting small singular values to reduce its dimension. Then the original word by context matrix is reformed from reduced dimensional space. Each sentence is represented in the form of a vector in the reduced dimensional space to compute sentence similarity. Then the similarity is measured by computing the distance between these vectors (measured, e. g. with cosine function). The limitation of this method is that the dimension is in a fixed size, so, input sentence will have a very sparse representation.

Rus et al. [43] use Latent Dirichlet Allocation (LDA) [44] to measure document /sentence similarity. LDA is a probabilistic approach to model a document into a distribution of topics. This method works by first semi-randomly assigning each word in a document by topics following Dirichlet distribution. This assignment makes each document is represented with topics, and each topic is represented by words. The method will conduct a repeated update of this assignment by considering the proportion of words in a document that are assigned to a topic and proportion of assignments to a topic, overall documents, that come from a word. This update will continue it converge to steady-state. As a result, we obtain a document representation in the distribution of topics and topics representation in the distribution of words. Topic distribution of a sentence is compared with the topic distribution of other document using Hellinger distance formula to measure document similarity.

A different approach is taken by Gabrilovich and Markovitch [45] by proposing the Explicit Semantic Analysis (ESA) method to measure the relatedness of the text fragment. The method represents text input into an ordered sequence of a weighted vector in a high-dimensional concept extracted from Wikipedia corpus. Then the semantic relatedness is calculated by comparing vector representation using distance metrics, for example, Cosine coefficient.

Shrestha [46] proposed a method based on the Vector Space Model (VSM). First, the method builds a term-document matrix with the document, as the dimension, is a training corpus and term is a unique term among the training corpus. However, unlike the regular VSM, it reduces the dimension by only kept the dimension with value 1. After the term vectors are obtained, then it is used to construct a document vector for the sentence being compared. The term vector is added to the sentence if the term is present. The method also adds a weighting scheme for Inverse Document Frequency to the document vectors.

Another approach is proposed by Kusner et al. [47]. They leverage word2vec technique by Mikolov et al. [48] to generates word embedding from Google News corpora. Word embedding means representing words into a dense numerical vector representation. The distance between the word embedded vector is semantically meaningful to a certain extent. The method represents two sentences into normalized Bag-of-Word vectors to measure the sentence similarity. The distance between the two sentences is measured using the Word Mover Distance (WMD) function. The function calculates the minimum cumulative distance that word in the first sentence needs to travel to match exactly the word in the second sentence. The distance between words is measured using Euclidean distance between that word embedded vectors. As the final result of WMD computation, the more distance two sentences have, the less similar the two sentences are.

3.2.6.4. Hybrid methods

Li et al. [52] proposed a method to calculate sentence similarity by considering semantic and word order information implied in the sentences. To calculate the semantic meaning of the sentences, it combines a knowledge-based and corpus-based method. The method combines two input sentences into joint word set. Then the input sentences are transformed into a raw semantic vector by using knowledge from a lexical database (WordNet) and joint word set as vocabulary. Each raw semantic vector component will be assigned value of one if it is present in the joint word set. However, if not, the degree of similarity between words will be calculated by considering the shortest path between the two words and the depth of subsumer in WordNet taxonomy. With a similar mechanism, order vectors are also constructed. Each word in a sentence has different significance to the meaning of the sentence; therefore, different weighting must be applied to each word. The method does this by using information content derived from a corpus (Brown corpus). Semantic vectors are formed by combining the raw semantic vector with this information content. Then semantic and order similarity is calculated for each respective vectors. Finally, sentence similarity is calculated by combining semantic and order similarity. This method drawback is that it does not consider word sense disambiguation, which can lead to inappropriate selection of sense and false word similarity calculation.

To overcome the problem in the method by Li et al. [52], Pawar and Mago [53] propose a method that is similar but extends is capability by adding word sense disambiguation steps. The method starts by partitioning the input sentences into a list of tokens (tokenization). After that, Part-of-speech tagging is applied for each token/word to labels them accordingly. A semantic vector is constructed for each sentence, which contains the word similarity value assigned to each word for every other word from the second sentence in comparison. The calculation of word similarity is done by utilizing WordNet as a semantic net. This calculation is measured by considering the shortest path length between words and depth of least common subsumer in WordNet as a hierarchy. This process of semantic vector construction also considers information content derived from WordNet as a corpus. The method calculates semantic similarity from these two semantic vectors. As an optional capability, word order vector can be formed to calculate word order similarity. Finally, sentence similarity is measured by combining semantic and word order similarity.

Unlike two previous methods, Islam et al.[54] use string similarity and corpus-based similarity. For string similarity, they combine three types of modified Longest Common Subsequence and give different weight to each type. They also use Second Order Co-occurrence PMI [55] for corpus-based similarity and word order similarity checking. Similar

testing environment as in Li et al. [52] research is used. As a result, the method could achieve a Pearson correlation coefficient of 0.853, which outperform Li et al.'s method.

Mihalcea et al. [56] calculate sentence similarity by aggregating the maximum similarity score between each word of one sentence with each word in the pair's sentence. Then the value is weighted by Inverse Document Frequencies values of each word with the help of British National Corpus. The similarity between words is calculated by combining all six concept similarity formula that has been explained in section 4.1.2. They conduct a test to the method by using a dataset of MSRP. The method could achieve an accuracy of 0.703.

Vu et al. [57] use a different approach to measure sentence similarity by combining Explicit Semantic Analysis (ESA) [45] with Recall-Orientated Understudy for Gisting Evaluation (ROUGE) [58]. ROGUE is lexical similarity measure that based on n-gram co-occurrence statistic. They compute sentence similarity with each method and then calculate the final similarity by using a linear combination and a tuning parameter. They test the method by using their own synthesized dataset from Wikipedia articles. The experiment result shows that it could achieve the highest Person correlation between human-annotated score and the method's score by the value of 0.8265.

In 2012, the Association for Computational Linguistics (ACL) held a Semantic Evaluation (SemEval) workshop focusing on the analysis of diverse semantic phenomena of text. One of the tasks in the workshop is Semantic Textual Similarity where participants can submit methods to measure the level of equivalence of two sentences in a semantic manner [18]. The top three methods use a similar approach in which they combine several metrics and use the result as features input for machine learning models. The methods are from Bär et al. [59], Šarić et al. [60] ,and Banea et al. [61]. The first method from Bär et al. [59] combines numerous measurement including string-based (i.e. Greedy String Tiling, Longest Common Substring, Longest Common Subsequence, n-grams), knowledge-based (i.e. Resnik measures [31] with Mihalcea aggregation function [56] to scale up to sentence level), corpus-based (i.e. Explicit Semantic Analysis with Wikipedia and Wiktionary as resource corpus) and two additional text expansion mechanism (i.e. Lexical Substitution System, Statistical Machine Translation). The second best method is from Šarić et al. [60] which comprises n-gram overlap, WordNet-Augmented overlap, weighted word overlap (with Google Books as a corpus), vector space similarity, shallow Named Entity Recognition, and number overlap. The WordNet-Augmented overlap is built upon word similarity measurement form Leacock and Chodorow [29] while the vector space similarity is utilizing distributional vector of each word from Latent Semantic Analysis. The result of each measurement is used as a feature of a regression model. The third method from Banea et al. [61] combines knowledge-based, corpus-based semantic similarity, and bipartite graph matching. The calculation result from each similarity measures is used as features for supervised machine learning technique specifically support vector regression.

In a specific domain such as biomedicine and law, there is also a need to measure sentence similarity. However, this task has its challenges where the sentence being compared contains many terms which are specific to that domain. Soğancıoğlu et al. [62] propose a method to measure sentence similarity in the biomedical domain. The method input the text into each sentence similarity comprises knowledge-based similarity (combined ontology), string similarity (q-gram), and corpus-based (paragraph vector). The result of each measurement is passed to the supervised regression model. The combined ontology measure used both WordNet as general-purpose ontology and Unified Medical Language System as a biomedical ontology to cover biomedical terms that might be excluded in WordNet. On the

other hand, for paragraph vector, the method utilizes PubMed comprises biomedical corpus to build the vector model.

3.2.6.5. Strengths and weaknesses of the STS methods

In this section, we elaborate more in the context of strengths and weaknesses of the methods so a comparison can be made. Seven aspects in two groups are used to examine the method's strengths and weaknesses. The first group is related to the data requirements. The first aspect of this category is the domain and language dependency. This aspect looks at whether the methods can be used for various languages or a specific domain. The second aspect is the semantic knowledge requirement. We check whether these methods require semantic knowledge such as WordNet to derive word-to-word similarity so that the methods could work well. The third aspect is whether the method requires the existence of a corpus to derive relations between words before the relationship can be used for sentence-level similarity measurement, e.g., LSA, LDA, word2vec. The fourth is training data requirements. This aspect will evaluate whether the methods require training and testing data consist of annotated sentence pairs to function properly.

The second group is related to semantic similarity, which consists of three aspects. The first aspect of being examined is semantic meaning. We check the capability of the methods to identify a high degree of similarity from two sentences that have a similar meaning but a different sequence of character or words. The last aspect we review is polysemy, which is a word that has several meanings depending on the context where the word is used.

We list the findings we obtained from the literature in Table 4. The seven aspects that we have described above are displayed column named independent domain and language columns, requires semantic knowledge, requires corpus, requires training data, semantic meaning, word order, polysemy. The checklist mark in column 6,7,8,9 means that the aspect is required by the methods while in column 10,11,12 mean that the capability is present in the methods.

Table 4. Strengths and weaknesses of short text similarity measurement methods

Study ID	Name	Citation	Type	Technique Used	Evaluation Metric	domain and language independent	requires semantic knowledge	requires corpus	requires training data	semantic meaning	word order	Poly semy	remark
(1)	(2)	(3)	(4)	(5)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)
S1	Longest Common Substring	[19]	String	character matching, suffix trees	-	✓	-	-	-	-	-	-	Cannot identify semantic meaning
S2	Longest Common Subsequence	[20]	String	character matching	-	✓	-	-	-	-	-	-	Cannot identify semantic meaning
S3	N-Gram	[21]	String	character matching, N-Gram of characters	-	✓	-	-	-	-	-	-	Cannot identify semantic meaning
S4	Jaccard coefficient	[22]	String	term matching	-	✓	-	-	-	-	-	-	Cannot identify semantic meaning
S5	Dice coefficient	[24]	String	term matching	-	✓	-	-	-	-	-	-	Cannot identify semantic meaning
S6	VSM TF-IDF	[26]	String	term matching, Vector Space Model (VSM), TF-IDF weighting, Cosine similarity	-	✓	-	-	-	-	-	-	Cannot identify semantic meaning
S7	Liu & Wang	[34]	Knowledge	JCN word similarity, concept hierarchical model	paraphrase detection task on MSRP dataset, accuracy = 0.72, precision = 0.738, recall = 0.902, F-measures = 0.8111	-	✓	-	-	✓	-	-	Less accurate than LDA
S8	LSS	[35]	Knowledge	Path word similarity, VSM, Cosine similarity	Sentence similarity task on Rubenstein and Goodenough sentence pair dataset, Pearson correlation coefficient = 0.807	-	✓	-	-	✓	-	-	faster but less accurate than LSA and STASIS
S9	Li, Li, Cai, & Han	[36]	Knowledge	Information Content, Cosine similarity on Noun and Verb vectors	Sentence similarity task on CMU dataset, accuracy = 0.893, precision = 0.868, recall = 0.925, f-measure = 0.896	-	✓	-	-	✓	✓	-	Using their own testing dataset, so the result is incomparable with other research in term of performance and accuracy
S10	Castillo & Cardenas	[37]	Knowledge	Resnik, Lin, JCN, Pirro word similarity, bipartite graph matching, Hungarian algorithm	Textual Entailment recognition task, accuracy=56.83%	-	✓	-	-	✓	-	-	Text similarity as part of Text Entailment recognition system
S11	Concept forest	[40]	Knowledge	concept forest, IDF	Document clustering on Reuters dataset, accuracy=80%	-	✓	-	-	✓	✓	-	Text similarity as part of Document clustering
S12	O'Shea	[41]	corpus	Latent Semantic Analysis (LSA)	Sentence similarity task on Rubenstein and Goodenough sentence pair dataset, Pearson correlation coefficient = 0.838	-	-	✓	-	✓	-	-	More accurate than LSS and STASIS but slower due to its sparse matrix,

Study ID	Name	Citation	Type	Technique Used	Evaluation Metric	domain and language independent	requires semantic knowledge	requires corpus	requires training data	semantic meaning	word order	Poly semy	remark
(1)	(2)	(3)	(4)	(5)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)
													computationally expensive.
S13	Rus, Niraula, & Banjade	[43]	corpus	Latent Dirichlet Allocation (LDA)	paraphrase detection task on MSRP dataset, accuracy = 0.733, precision = 0.771, recall = 0.852, F-measures = 0.81	-	-	✓	-	✓	-	✓	polysemous words are contained in different topics
S14	Gabrilovich & Markovitch	[45]	corpus	Explicit Semantic Analysis (ESA)	Sentence similarity task on ABC news mail dataset, Pearson correlation coefficient = 0.72	-	-	✓	-	✓	-	✓	Using their testing dataset, so the result is incomparable with other research in term of performance and accuracy
S15	SVSM	[46]	corpus	Vector Space Model, Inverse Document Frequency	paraphrase detection task on MSRP dataset, accuracy = 0.683, precision = 0.703, recall = 0.917, F-measures = 0.796	-	-	✓	-	✓	-	-	Less accurate than LDA
S16	Kusner, Sun, Kolkin, & Weinberger	[47]	corpus	word2vec, Earth Mover Distance	document classification on 8 dataset, average kNN error 0.42	-	-	✓	-	✓	-	-	Text similarity as part of Documents classification
S17	STASIS	[52]	hybrid	words similarity based on WordNet, Information Content, word order similarity, and Cosine similarity.	Sentence similarity task on Rubenstein and Goodenough sentence pair dataset, Pearson correlation coefficient = 0.816	-	✓	✓	-	✓	✓	-	More accurate than LSS
S18	Pawar & Mago	[53]	hybrid	words similarity based on WordNet, Word Sense Disambiguation, Information Content, word order similarity, and Cosine similarity.	Sentence similarity task on Rubenstein and Goodenough sentence pair dataset, Pearson correlation coefficient = 0.875	-	✓	✓	-	✓	✓	✓	More accurate than STASIS
S19	STS	[54]	hybrid	Longest Common Subsequence, SOC-PMI, word-order similarity	Sentence similarity task on Rubenstein and Goodenough sentence pair dataset, Pearson correlation coefficient = 0.853	-	✓	✓	-	✓	✓	-	More accurate than STASIS
S20	Mihalcea	[56]	hybrid	PMI-IR, LSA, LCH, JCH word similarity, Inverse Document Frequency	paraphrase detection task on MSRP dataset, accuracy = 0.703, precision = 0.696, recall = 0.977, F-measures = 0.813	-	✓	✓	-	✓	-	-	Less accurate than LDA
S21	RESA	[57]	hybrid	ROGUE, ESA, N-Grams	Sentence similarity task on Wikipedia dataset, Pearson correlation coefficient = 0.8265	-	-	✓	-	✓	-	✓	outperform ESA on Person correlation coefficient

Study ID	Name	Citation	Type	Technique Used	Evaluation Metric	domain and language independent	requires semantic knowledge	requires corpus	requires training data	semantic meaning	word order	Poly semy	remark
(1)	(2)	(3)	(4)	(5)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)
S22	UKP	[59]	hybrid	Longest Common Subsequence, Longest Common Substring, n-grams, ESA, JCH, Lin, Resnik similarity, linear regression classifier	paraphrase detection task on MSRP dataset, Pearson correlation coefficient= 0.823	-	✓	✓	✓	✓	✓	-	Best performing method on SemEval 2012
S23	TakeLab	[60]	hybrid	knowledge and corpus-based (LSA) word frequencies	paraphrase detection task on MSRP dataset, Pearson correlation coefficient = 0.813	-	✓	✓	✓	✓	✓	-	Second best performing method on SemEval 2012
S24	UNT	[61]	hybrid	WordNet-based word similarity, LSA, ESA, SSA, bipartite graph matching, vector regression	paraphrase detection task on MSRP dataset, Pearson correlation coefficient = 0.785	-	✓	✓	✓	✓	✓	✓	Third best performing method on SemEval 2012
S25	BIOSSES	[62]	hybrid	knowledge-based word similarity, Q-gram, word2vec, regression model	Sentence similarity task on Biomedical TAC dataset, Pearson correlation coefficient = 0.836	-	✓	✓	✓	✓	✓	-	Biomedical domain

3.2.6.6. Semantic knowledge and corpus resource

Knowledge-based and corpus-base methods require the semantic meaning of words and relations between words obtained from the semantic knowledge and corpus resource to function in measuring sentence similarity. Semantic knowledge takes the form of a network that consists of words or concepts that are interconnected with an explicit relationship such as is-a or part-of. Meanwhile, the corpus is only a collection of documents consisting of words that the relationships have not been explicitly known. Certain techniques that combine statistics, probabilities, and neural networks such as word2vec, LSA, and LDA, can be used to derive relationships between words or documents.

In the literature, we find numerous semantic knowledge and corpus resources, which can be classified into two categories, namely general-purpose and domain-specific resource. The former category means that the resources contain a larger proportion of common words that do not come from specific domains such as dictionaries, newspaper articles, non-scientific books. Whereas, the latter category is the opposite where resources mostly contain words or concepts from specific domains such as biomedicine, chemistry, law. General-purpose semantic knowledge and corpus are widely used in research to develop general semantic text semantic measurement methods and not tied to a specific domain.

For semantic knowledge resource, we find a general-purpose resource, namely WordNet and two domain-specific resources, namely the Unified Medical Language System (UMLS) and the Chemical Entities of Biological Interest (ChEBI). WordNet is a lexical ontology in the English language which was launched by Princeton University in 1985. This resource contains words or concepts, along with definitions similar to dictionaries. Each word that has the same meaning is grouped in synsets. WordNet contains 17,000 synsets that are connected to a relationship forming a semantic network [27]. Various versions of WordNet for other languages are also available such as HowNet for Chinese [63], [64]. In the field of biomedicine, UMLS has widely used for text semantic similarity, which extensively uses many specific words or terms in that field [65]. UMLS itself is an ontology which contains concepts in fields that are a combination of several ontologies such as SNOMED-CT, MeSH, and Gene Ontology. UMLS was launched in 1986 and has been hosted by the US National Library of Medicine ever since. In the field of chemistry, ChEBI is often used for text semantic similarity measurement because it contains many concepts from domain chemical compounds such as molecular entities, alkanes, alkyl groups [66]. ChEBI has been developed by the European Bioinformatics Institute since 2002.

In term of corpus resource, we find research papers that use general-purpose corpus such as Wikipedia [45], British National Corpus (BNC) [56], Brown corpus [52], [53] and corpus from various news articles such as Google News [47], New York Times [60] and Reuters [67]. BNC is a collection of 100 million words gathered from books, newspaper articles, journals and essays and various kinds of writing. BNC was launched in 1994 and maintained by a consortium led by Oxford University [68]. Meanwhile, Brown Corpus is a collection containing 1,014,312 American English words from various sources such as novels, journals, articles and from various categories including fiction, religion, government, social, and political[69]. This Corpus was published by Kucera and Francis of Brown University in 1961. For the specific corpus domain, we found two corpora, PubMed and FindLaw. PubMed is a corpus containing a collection of articles, books, and journals of biomedicine compiled by the United States National Library of Medicine since 1966. Each year new documents are added selectively so that by 2018, this corpus has 29.1 million records. The corpus is widely used for text semantic

similarity in the field of biomedicine, such as in [62], [70]. Whereas FindLaw is a collection that contains 35,000 legal case documents that were crawled from the FindLaw site. This corpus is used for semantic similarity measurement in the field of law [71]. In summary, we list all of our findings in Table 5.

Table 5. Semantic knowledge and corpus resource

No	Name	Type	Domain	Description
1	WordNet	semantic knowledge	General	Lexical ontology for the English language
2	HowNet	semantic knowledge	General	Lexical ontology for the Chinese language
3	UMLS (SNOMED-CT)	semantic knowledge	Biomedical	Biomedical ontology
4	ChEBI	semantic knowledge	Chemical	Chemical compound ontology
5	Wikipedia	corpus	General	English articles
6	New York Times	corpus	General	news corpus
7	Google News	corpus	General	news corpus
8	Reuters	corpus	General	news corpus
9	British National Corpus	corpus	General	English document from various source and category
10	Corpus Brown	corpus	General	English document from various source and category
11	PubMed	corpus	Biomedical	Biomedical documents from journal, books, article
12	FindLaw	corpus	Law	Law case documents

4. EWC Code Recommender System

We obtain insight about methods of measuring text similarity and resource that are commonly used from the SLR that has been conducted. This insight becomes the basis for designing the EWC Code Recommender system.

4.1. EWC Recommender System Model

As discussed in the Problem Definition section, the research problem is how to accurately give EWC code recommendation whenever the user inputs a waste item description. We build a model which can be seen in Figure 12 to solve this problem. There are two types of data input, which are Industrial Symbiosis (IS) Data and EWC Data. The former contains all the waste item that is registered to the Sharebox platform, including its description, while the latter contains EWC codes and its description. For each waste item, we look for relevant EWC codes in the EWC code catalog by comparing waste item description with each of EWC codes entry description. Preprocessing steps like tokenization, stemming, and lemmatization using NLP is applied to the descriptions before further preprocessing. We build three types of RS, which are string-based RS, knowledge-based, and corpus-based recommender systems. String-based RS doesn't have the capability to comprehend the semantic meaning of a text, and we use it as a baseline. Semantic awareness is incorporated in the last two RS, and the effect will be observed. Each system generates a list of EWC codes recommendation for a certain waste item. EWC codes which description is within the top N most

similar list is returned as recommendations. We describe each part of the system in the following section. The source code of the EWC Code RS is available online on GitHub[2].

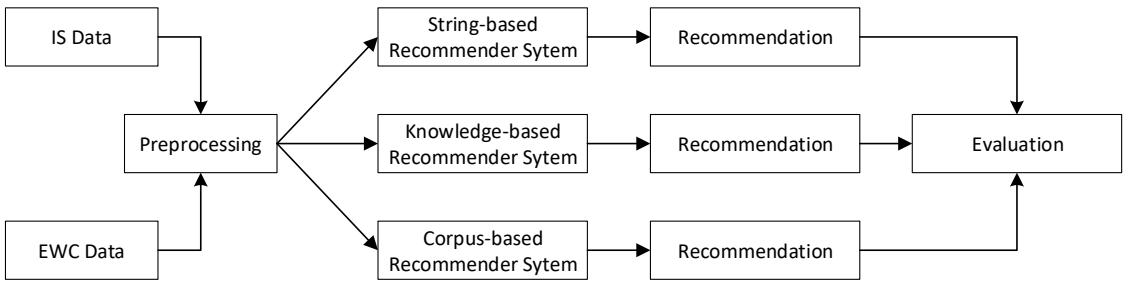


Figure 12. EWC code Recommender System model

4.2. Dataset

4.2.1. Industrial Symbiosis (IS) dataset

IS dataset contains all waste item that is similar to data that the user input to the Sharebox platform. It is collected through a workshop setting from the industrial cluster in Europe. It comprises five fields, as shown in Table 6. The ID field is a waste item ID. The Company ID field is an identifier of the Sharebox user. We anonymize the company name to maintain the privacy of the user. Waste Description is a short text description of the waste item. Type is the type of the waste item. EWC code field is the manually labeled EWC code by the user. We consider this as a correct code since the user is an expert who works in Industrial Symbiosis field. We use this label to validate the recommended EWC codes. In total, there are 746 rows. But for the recommendation process, we only use data with type Material and labeled with valid EWC code (not NULL) which are 311 rows or 48% of the entire IS dataset.

Table 6. Industrial Symbiosis (IS) dataset

ID	Company ID	Waste Description	Type	EWC code
1	1	Lab: Solid waste with the relevant laboratories in existing GC-MS elemental analysis, calorimetry , oxygen permeability , extruders, including laboratory analysis on the FCP- MS instrument can be analyzed to support research projects . (A part of the Lab is accredited)	Service	NULL
2	1	Waste and hazardous waste management: Waste and hazardous waste management, management of AEEE , sludge decision support systems , life cycle analysis, etc. It can be carried out joint projects with companies on issues .	Service	NULL
3	2	plastic chips: Trimmed , broken scrap plastic	Material	15 01 02
4	2	Pressed scrap metal	Material	15 01 04

[2] https://github.com/dimaswprakoso/EWC_string_based, https://github.com/dimaswprakoso/EWC_knowledge_based, https://github.com/dimaswprakoso/EWC_corpus_based

ID	Company ID	Waste Description	Type	EWC code
5	2	Pressed scrap paper: Waste recycling facilities for paper mills or considered as intermediate products or raw materials	Material	NULL
…	…	…	…	…

The distribution of human-annotated EWC codes contained in IS dataset can be seen in Table 7. From 841 EWC codes available in the catalog, only 120 codes are used by 356 valid rows in IS dataset, so, many EWC codes are not represented in the IS dataset. Of the 120 codes, more than half of it (58%) have a frequency of 1 row only.

Table 7. The distribution of human-annotated EWC Code on IS dataset

No	EWC code	Freq	No	EWC code	Freq	No	EWC code	Freq	No	EWC code	Freq
1	15 01 01	21	31	17 09 03	3	61	15 01 99	1	91	02 06 99	1
2	12 01 03	15	32	15 01 10	3	62	02 02 03	1	92	17 05 04	1
3	12 01 10	13	33	17 02 01	3	63	10 09 99	1	93	10 06 99	1
4	19 08 99	11	34	01 04 09	3	64	16 10 04	1	94	16 05 06	1
5	17 09 04	10	35	17 01 03	2	65	17 04 05	1	95	19 06 06	1
6	15 01 02	9	36	10 09 08	2	66	07 06 04	1	96	19 12 04	1
7	15 01 06	9	37	13 02 05	2	67	16 06 05	1	97	21 01 08	1
8	15 01 03	9	38	07 02 13	2	68	01 01 02	1	98	02 01 01	1
9	20 03 01	9	39	15 01 04	2	69	08 02 02	1	99	08 01 11	1
10	12 01 99	7	40	16 01 07	2	70	07 06 12	1	100	11 03 02	1
11	03 01 01	7	41	20 01 21	2	71	20 01 99	1	101	10 07 99	1
12	17 01 07	6	42	15 01 05	2	72	08 02 03	1	102	10 13 06	1
13	16 01 17	5	43	16 11 06	2	73	10 01 26	1	103	06 10 02	1
14	02 03 04	5	44	10 13 14	2	74	04 02 22	1	104	02 01 99	1
15	12 02 16	5	45	17 04 07	2	75	11 01 08	1	105	10 12 11	1
16	10 02 02	4	46	15 01 11	2	76	06 03 11	1	106	16 07 08	1
17	06 13 99	4	47	17 08 02	2	77	10 01 99	1	107	10 01 01	1
18	02 01 03	4	48	03 01 99	2	78	17 03 03	1	108	13 05 08	1
19	12 01 01	4	49	03 01 05	2	79	11 01 09	1	109	20 01 39	1
20	15 02 02	4	50	02 02 99	2	80	05 01 06	1	110	20 01 35	1
21	17 04 01	4	51	16 06 02	2	81	06 10 99	1	111	06 01 06	1
22	12 01 05	4	52	17 09 99	1	82	12 01 14	1	112	20 01 40	1
23	17 02 03	4	53	12 01 06	1	83	06 03 13	1	113	17 06 03	1
24	08 01 12	4	54	20 01 25	1	84	12 01 04	1	114	19 02 05	1
25	20 01 01	4	55	16 06 01	1	85	17 04 02	1	115	17 03 02	1
26	16 02 16	3	56	16 03 05	1	86	10 02 01	1	116	13 02 08	1
27	19 12 12	3	57	10 09 03	1	87	01 01 01	1	117	19 02 03	1
28	17 01 06	3	58	13 01 99	1	88	14 06 03	1	118	10 13 01	1
29	16 02 14	3	59	10 12 08	1	89	02 05 02	1	119	16 03 03	1
30	02 01 04	3	60	16 05 04	1	90	20 01 15	1	120	20 03 07	1

4.2.2. EWC dataset

EWC is a waste codification standard published in 2005 by the European Commission. Industrial or household waste must be labeled with EWC code if it will be circulated in the European Union area. EWC consists of 6 digit numbers that form a hierarchy. Figure 13 illustrates the hierarchy of an EWC code example.

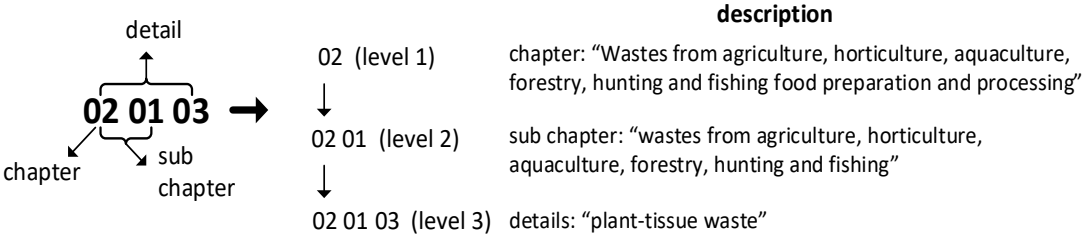


Figure 13. EWC code structure

The first two digits are the top-level that explains the chapter or level 1. The next two digits are a more detailed level of subchapters or level 2, and all six digits show details or level 3 [72]. Each level has its own description. The higher the level, the broader the context is. In this example, the level 3 EWC code is 02 01 03 with description "plant-tissue waste". Its ancestor is level 02 with code 02 01 and description "wastes from agriculture, horticulture, aquaculture, forestry, hunting, and fishing." This level has broader context that covers all the waste in that category including plant-tissue waste. The level 1 of this example is code 01, which is even more general description. Code 01 covers all the wastes from agriculture, horticulture, aquaculture, forestry, hunting, and fishing food preparation and processing. In total, there are 20 codes for level 1, 111 codes for level 2, and 841 codes for level 3. Table 8 show some excerpts of the EWC dataset for each level. The complete list of EWC codes is available online on Pureplanet website[3] . This thesis focuses on level 3 EWC code description only.

Table 8. EWC codes example

Level 1 Code	Description
01	Wastes resulting from exploration, mining, quarrying, physical and chemical treatment of minerals
02	Wastes from agriculture, horticulture, aquaculture, forestry, hunting, and fishing food preparation and processing
03	Wastes from wood processing and the production of panels and furniture, pulp, paper and cardboard
…	…

Level 2 Code	Description
01 01	wastes from mineral excavation
01 03	wastes from physical and chemical processing of metalliferous minerals
01 04	wastes from physical and chemical processing of non-metalliferous minerals
…	…

Level 3 Code	Description
01 01 01	wastes from mineral metalliferous excavation
01 01 02	wastes from mineral non-metalliferous excavation
…	…

We made statistical analysis from a joint set of waste item description from IS dataset and EWC code description from EWC data. After stop word removal, the dataset characteristic can be

[3] https://www.pureplanetrecycling.co.uk/list-of-waste/

seen in Figure 22 and 23. Figure 14 shows sentence length in the dataset varies from 1 to 22 words, but the average is only 4.83 long. In such a short text, word co-occurrence might low. If we apply methods that require a high number of co-occurring words to measure text similarity, then it may perform poorly. Figure 15 shows the most frequent words that appear in the dataset. We omit frequent but common words such as *waste* and *wastes* since it gives no contribution or even introduces error in the short text similarity measurement as part of the recommendation process.

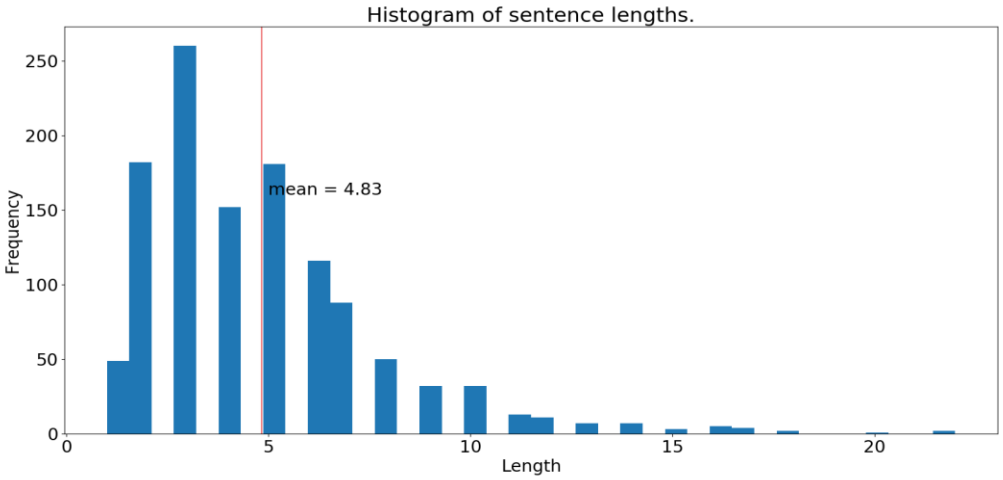


Figure 14. The length of the waste item and EWC description

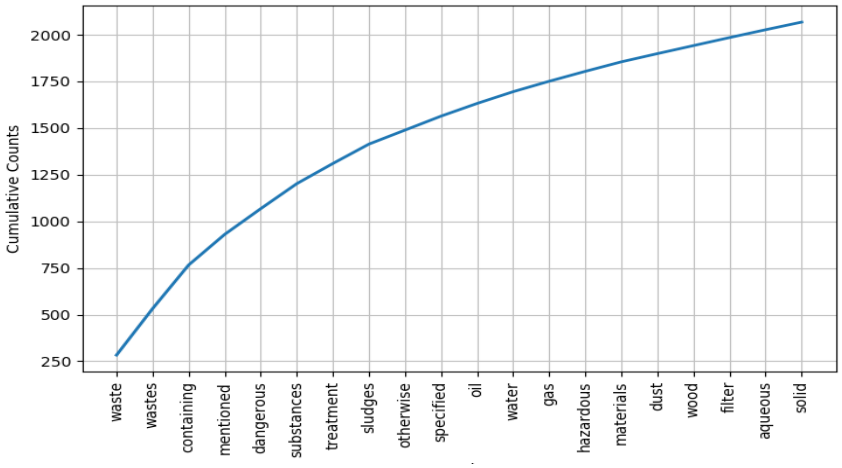


Figure 15. The most frequent words in the dataset

4.3. Data Preprocessing

The initial stage in each recommendation process is text preprocessing. It comprises four steps.

1. **Lower case**: the waste item and EWC code description are converted to lower case.
2. **Tokenization**: the words in the description are separated, number and punctuation are removed.
3. **Short word removal**: words with a length of fewer than three characters are removed.
4. **Stop word removal**: English stop word such as 'the', 'is', 'are' is removed. We utilize stop word in NLTK python library to provide the stop word list.

5. **Exclude common word**: words that are frequent but common in the dataset such as word *waste* and *wastes* from Figure 22 are removed. We also add other words that are common in the Industrial Symbiosis field. The complete list of excluded words can be seen in Table 9.

Table 9. Excluded term list

No.	Term	No.	Term	No.	Term	No.	Term	No.	Term
1	based	11	material	21	product	31	solution	41	unused
2	consultancy	12	materials	22	production	32	solutions	42	use
3	consumption	13	organic	23	quantity	33	specific	43	used
4	containing	14	other	24	recycle	34	specified	44	waste
5	dangerous	15	otherwise	25	recycling	35	spent	45	wastes
6	hazardous	16	particular	26	remainder	36	substance		
7	industrial	17	process	27	residue	37	substances		
8	industry	18	processed	28	scrap	38	support		
9	management	19	processes	29	scraps	39	training		
10	managing	20	processing	30	service	40	unprocessed		

6. **Stemming and lemmatization:** stemming and lemmatization is a process of deriving a term into its base form by removing the inflection. The difference is that stemming may result in a word that is not a dictionary word. We use Porter stemmer and word lemmatizer from NLTK python library.

4.4. Non-Semantic Aware EWC Recommender System

4.4.1. String-based EWC Code RS (baseline)

As a baseline, we develop a string-based EWC code RS by using the Vector Space Model with Term Frequency weighting approach from Salton [26]. This RS relies on word co-occurrence between the waste item description and EWC code description. If there is no shared term, then the EWC code will not get recommended even if its EWC description is semantically similar to the waste item description.

We design the string-based EWC Code RS, which is illustrated in Figure 16. The input of the system is IS data and EWC data which contain waste item and EWC codes, respectively. Both are accompanied by their description. The detail of the dataset is explained in section 4.2.

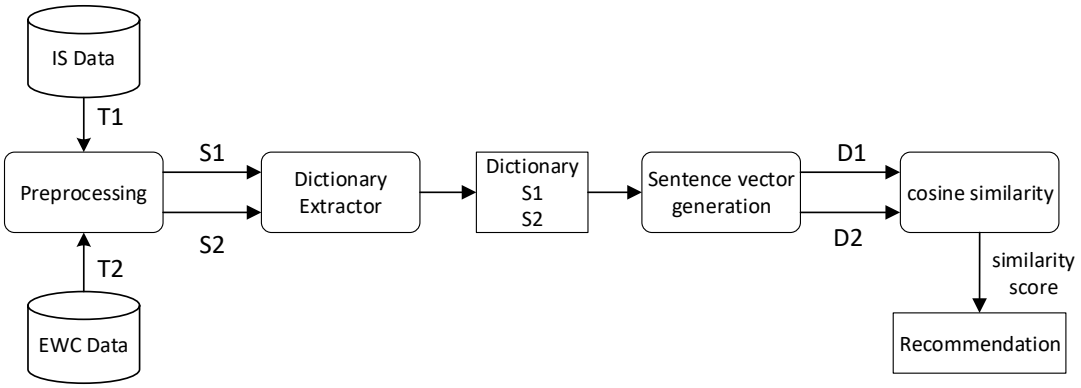


Figure 16. String-based EWC code recommendation system model

The method works as follows. For each waste item description T1 contained in IS Data, the method will scan the EWC data to find relevant EWC code by comparing each EWC code description T_2 with T_1. Both T_1 and T_2 are preprocessed using NLP techniques such as tokenization, stemming, and lemmatization, as described in section 4.3. The result is a bag of word representation S_1 for T_1 and S_2 for T_2. A dictionary is formed by joining unique words contained in S_1 and S_2:

$$D = S_1 \cup S_2 = \{w_1, w_2, w_3, \dots, w_n\}$$

The method transforms S_1 and S_2 to sentence vectors named D_1 and D_2 by using D as the vector dimension. Each element of the sentence vector corresponds to a word in D. The vector element value is set by referring to a rule as follows. If w_i present in S_1, then the *i*-th element of D_1 or D_{1i} is set to w_i occurrence frequency on S_1 (Term Frequency or TF). If it is otherwise, the element is set to zero. We also apply the IDF weighting mechanism from Salton and Buckley [26] to give higher importance to a word that is rarely occurred. It can be calculated by using equation (8) where n denotes the number of the sentences, $df(w_i)$ denotes the frequency of $w_i$occurrence in all sentences.

$$IDF(w_i) = \ln\left[\frac{n+1}{df(w_i)+1}\right] \tag{8}$$

After semantic vector D_1 and D_2 are obtained, we measure the sentence similarity score S_s of T_1 and T_2 by calculating the cosine coefficient of its sentence vectors using equation (9). Value of 0 means both sentences are dissimilar while a value of 1 means otherwise. The process will be repeated for each EWC code description in the catalog, and the result is saved for the recommendation process.

$$S_S = \frac{D_1 \cdot D_2}{||D_1|| \cdot ||D_2||} = \frac{\sum_{i=1}^{n} D_{1i} D_{2i}}{\sqrt{\sum_{i=1}^{n} {D_{1i}}^2}\sqrt{\sum_{i=1}^{n} {D_{2i}}^2}} \tag{9}$$

The recommendation process is conducted by sorting all the sentence similarity score in descending order. The system will return the EWC codes whose similarity score is within the Top-N list. We also incorporate sentence similarity threshold to limit the recommendation. However, we set it very low at 0.0001 so that even if there is only one similar word pair in the description pair, then the associated EWC code will still be considered as a candidate for a recommendation. To measure the system performance, we repeat the process for each waste item contained in IS data. The performance is measured by the method explained in section 4.6.

Worked-example

For example, given T_1 and T_2 as follows.

- T_1: *cardboard, wood and paper scrap*
- T_2: *paper and cardboard packaging*

The method applies the preprocessing step and obtains a bag of word representation of T_1 and T_2. Word *and* has been removed as the result of stop word removal.

- S_1: {*cardboard, wood, paper, scrap}*
- S_2: {*paper, cardboard, packaging}*

After the preprocessing step, the method extracts dictionary D by joining unique words from both S_1 and S_2.

- D: {*cardboard, wood, paper, scrap, packaging*}

The method then generates sentence vector D_1 for S_1 and D_2 for S_2, as illustrated in Figure 17. We use S_1 as an example and apply the same process to S_2. The method will do a string matching for w_i with each term in S_1. If it found a match, then it will save the accumulation as term frequency (TF). The figure uses the word *cardboard* as an example and found only one exact match. The TF is used as the value of the element of sentence vector D_1 for that corresponding w_i.

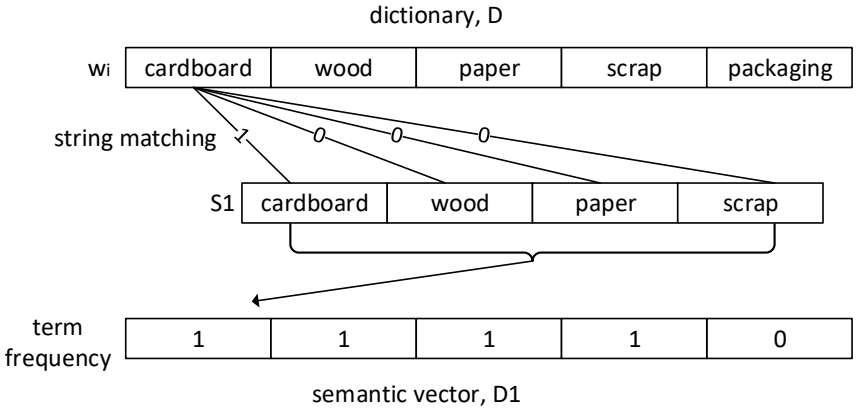


Figure 17. Sentence vector generation in SB EWC Code RS

- D1 = {1 1 1 1 0}
- D2 = {1 0 1 0 1}

The method calculate the IDF for each term in dictionary by using equation (8).

- IDF(cardboard) = ln[(2+1)/(2+1)]+1 = 1
- IDF(wood) = ln[(2+1)/(1+1)]+1 = 1.4055
- IDF(paper) = ln[(2+1)/(2+1)]+1 = 1
- IDF(scrap) = ln[(2+1)/(1+1)]+1 = 1.4055
- IDF(packaging) = ln[(2+1)/(1+1)]+1 = 1.4055

IDF weighting mechanism is applied to D_1 and D_2 by multiplying each vector element with the IDF value of its corresponding word in dictionary.

- D1 = {1 1.4055 1 1.4055 0}
- D2 = {1 0 1 0 1.4055}

Finally, the similarity of S_1 and S_2 is measured by calculating the cosine coefficient of its sentence vector D_1 and D_2 using equation (9). We get the sentence similarity score of 0.4112. The recommendation process for a single item follows this process. The similarity of a waste item description is compared with each EWC Code description. The score is saved and sorted in descending order for the recommendation process.

4.5. Semantic-Aware EWC Code Recommender System

In the baseline model, semantically similar words are ignored and does not contribute to the calculation of sentence-level similarity. To improve this, we design RS that has semantic-awareness which can comprehend the semantic meaning of a sentence. With this capability, the method is able to identify sentences with similar meaning even if the sentences are composed of different string. We incorporate semantic-awareness to the RS by using two approaches. The first one is knowledge-based, which leverage WordNet as a knowledge base for word similarity method. The second one is a corpus-based method that utilized word embedding generated by word2vec method to determine word relatedness. The details of the two approaches are described in more detail in the following sections.

4.5.1. Knowledge-based EWC Code RS

The EWC code recommender works by comparing the similarity between the description of a waste item with the description of all EWC code entry in the EWC code catalog. For each pair of the descriptions, the similarity is determined by aggregating the similarity between words that make up the pair. Therefore, we need a method to obtain the similarity between words not only in lexical but also in a semantic way.

4.5.1.1. Word similarity method

Knowledge-based word similarity methods utilize a network of concepts/terms that are semantically interrelated to extract similarity. The semantic network is varied and can be specific to certain domains such as biomedicine, law, or general-purpose semantic networks such as WordNet. WordNet is a lexical ontology that is similar to a dictionary that contains the concepts or words, and its definition [27]. Every concept or word that has the same meaning is grouped in synonym set or synset. Each synset is connected in a relationship that forms a semantic network/taxonomy. The relationships can be in the form of *a-part-of*, *a-kind-of*, *is-the-opposite-of*. WordNet also assigns the word type to each synset, which is a noun, verb, adjective, or adverb.

In the literature, we can find numerous formula to measure the degree of relatedness between the concept in the semantic network. The Path algorithm [28] consider the maximum depth of the concepts being compared and the path length in the taxonomy of the semantic network. It uses the equation (10) to measure the concept similarity. Wu and Palmer (WP) [30] includes another factor which is the depth of Least Common Subsumer (LCS) into the calculation as formalized in equation (11). LCS is the most specific ancestor in which the concepts have in common. Lin [32] consider the same factor in his calculation but use a different formula as defined in equation (12). Li [52] proposes another word similarity calculation by using equation (13) where l is the shortest path length between word or concept c_1 and c_2, h is the depth of LCS in the semantic network. α and β are constants that are determined through experiment. In WordNet, Li found that the optimal parameter α is 0.2 and β is 0.45.

$$Sim_{Path}(c_1,c_2) = (2 * depth_{max}) - len(c_1,c_2) \quad (10)$$

$$Sim_{WP}(c_1,c_2) = \frac{2 * depth(LCS(c_1,c_2))}{depth(c_1) + depth(c_2)} \quad (11)$$

$$Sim_{Lin}(c_1,c_2) = \frac{2 * IC(LCS(c_1,c_2))}{IC(c_1) + IC(c_2)} \quad (12)$$

$$Sim_{Li}(c_1, c_2) = e^{-\alpha l} * \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \quad (13)$$

In designing EWC Code recommender, we use WordNet as our knowledge base and metric Path, WP, Lin, and Li as a word similarity measurement method. However, in WordNet, a relation exists only between synset, not words. Therefore, when we try to measure word similarity between word pair, we query all the synset for each word first, then we apply word similarity metric to the synset pair. We select the maximum similarity of synsets pair and set it as a similarity value for both words. We provide an illustration in Figure 18. For example, we try to measure the similarity between words "*metal*" and "*iron*" using WP similarity. If we query the synsets for both words in WordNet, *metal* has two synsets while *iron* has four synsets. We use only the synsets with word type of noun to keep the example simple. We calculate the similarity of all possible synset pair. Then we select the maximum similarity and assign it as word similarity score for words *metal* and *iron,* which is 0.9333.

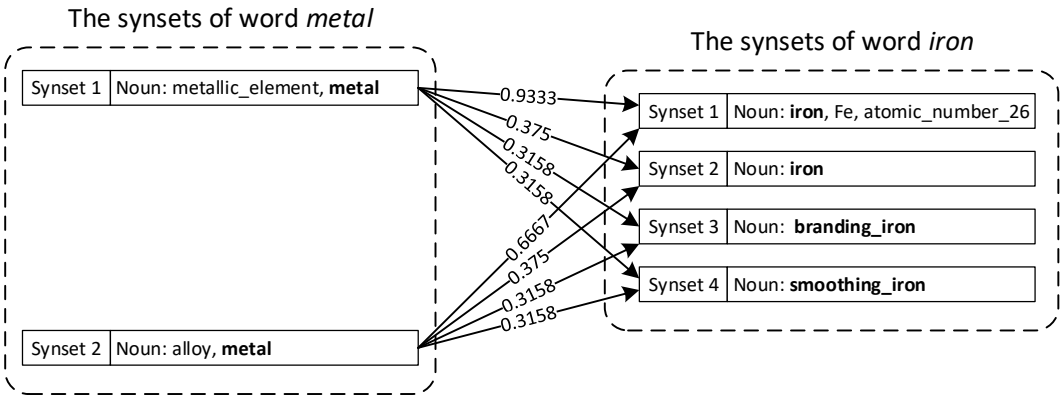


Figure 18. Example of maximum similarity selection from all synset pair

One important aspect in WordNet is that a concept has word type or part of speech label. The supported types are noun, verb, adjective, adjective satellite, and adverb. However, the relation between concept only appears for noun and verb. Therefore, the concept similarity method using equation (10) to (11) only applicable for that word types.

4.5.1.2. Short-text similarity method and recommendation process

We design knowledge-based EWC code RS by utilizing the word similarity method explained in the previous section. Figure 19 shows the model of the recommender system. The input of the system is IS data and EWC data which contain waste item and EWC Code catalog, respectively. Both are accompanied by their description. The detail of the dataset is explained in section 4.2.

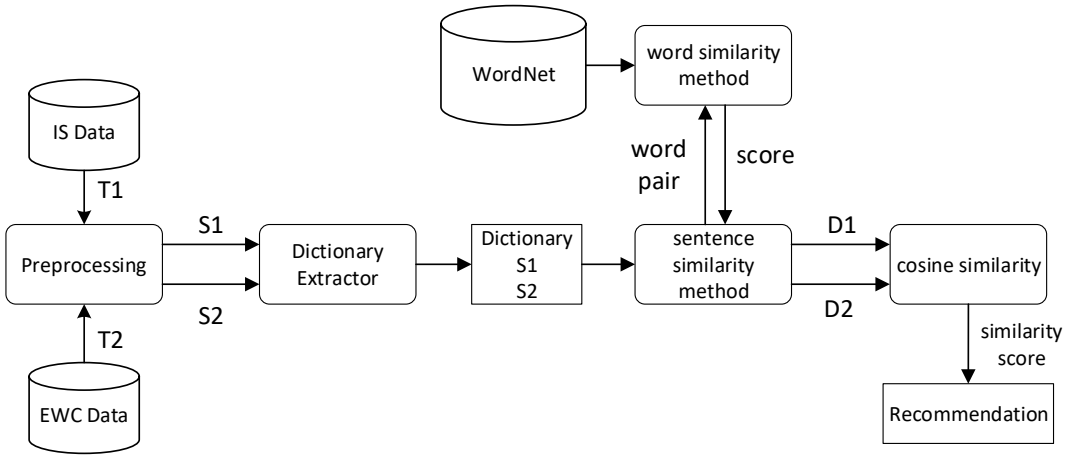


Figure 19. Knowledge-based EWC code recommendation system model

The method works similarly to string-based RS in term of data input and generation of S_1 and S_2. The difference occurs in the sentence similarity measurement process. KB methods conduct this by referring to the technique from Li [52]. It transforms S_1 and S_2 to sentence vectors named D_1 and D_2 by using D as the vector dimension. Each element of the sentence vector corresponds to a word in D. The vector element value is set by following a rule. We use S_1 as an example.

- If w_i present in S_1, then the *i*-th element of D_1 or D_{1i} is set to 1.
- If w_i does not exist in S_1, then the vector element value will be set to the maximum value of word similarity score between w_i and all the words in S_1. If the score is exceeding a preset word similarity threshold, then the vector element value will be set to this value; otherwise 0. The word similarity itself is calculated using the method described in section 4.5.1.1. The purpose of the word similarity threshold usage is that we want to eliminate words with low similarity since it will introduce noise to the sentence calculation. The threshold value is carefully set through experiment. If it is too low, it will introduce noise, while if it is too high, important word pairs could be skipped.

After semantic vector D_1 and D_2 are obtained, we measure the sentence similarity score S_s of T_1 and T_2 by calculating the cosine coefficient of its sentence vectors using equation (9). The remaining process is conducted similarly as SB EWC Code RS, including the recommendation process.

Although our sentence similarity method refers to the study by Li [52], we made several adjustments to improve the performance. First, we include preprocessing step such as word filtration by excluding stop word and word that are common in IS field. The reason is that those words give no contribution to similarity calculation. In the original version, they use all the words as it appears in the sentence. Second, we use maximum similarity for synset pair. We think this is more appropriate because it mimics human behavior that tends to maximize pattern-seeking. In the original version, they use that first synset pair that occurred in the WordNet. Third, we handle words that are not covered in WordNet such as names, technical terms, and miss-spelled words or typos. We calculate word similarity using Ratcliff/Obershelp pattern recognition[73]. Li's method just set word similarity to 0 if the term is not found in WordNet.

Worked-example

Let us consider waste description T_1 and EWC Code description T_2 from IS Data and EWC data, for example as follows. Both sentences have no shared word. If we apply string-based sentences similarity, the result will be dissimilar because both sentences are orthogonal in vector space.

- T_1: *iron and steel scrap*
- T_2: *ferrous metal*

After the preprocessing step, the method obtains S_1 and S_2 as a bag of word representation of the sentences. We can see that the word *and* is removed because of the stop word removal in the preprocessing step.

- S_1: {*iron, steel, scrap}*
- S_2: {*ferrous, metal}*

Then the method extract dictionary D using dictionary extractor.

- D: {*iron, steel, scrap, ferrous, metal*}

By using S_1, S_s, and D, the method generates sentence vectors as illustrated in Figure 20. Let us take S_1 as an example.

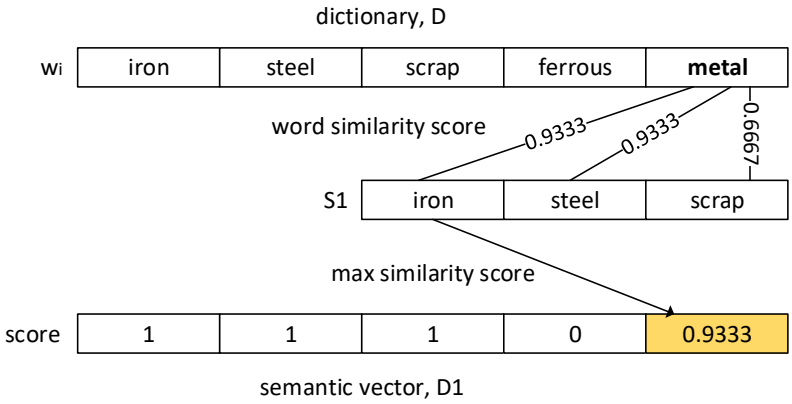


Figure 20. Sentence vector generation in KB EWC Code RS

The method generates semantic vector D_1 for S_1 by checking the presence of each corresponding dictionary word wi in S_1. For example, word *iron, steel,* and *scrap* are contained in S_1; therefore, the first, second, and third element of D_1 is set with value 1. Word *metal* doesn't exist in S_1, so the method calculates the word similarity between w_i and each word in S_1 using the method described in section 4.5.1.1. This example uses the WP algorithm as a synset similarity method. We can see that word *iron* has the maximum word similarity score; therefore, the fifth element of D_1 is set by 0.9333. By using the same mechanism, we generate semantic vector D_2.

- D1 = {1 1 1 0 0.9333}
- D2 = {0.9333 0.9333 0.6667 1 1}

To measure the sentence similarity, we calculate the cosine coefficient of both semantic vectors using equation (9). We get the value of 0.8611, which means both words is highly

similar even though there are no shared words. The result matches human inference where the sentence *"iron and metal scrap"* and "*ferrous metal"* are indeed similar to a certain extent. The string-based method would be blind to this kind of similarity. It would return the cosine coefficient of 0 because it finds no word string match between words in S_1 and S_2.

4.5.2. Corpus-based EWC Code RS

Figure 21 depicts the corpus-based EWC code RS model. It incorporates word2vec, which is a technique introduced by Mikolov et al. [48] to extract word embedding from a large corpus (Google news in our case). Word embedding is a representation of words in a dense numerical vector. The distance between the word embedded vector is semantically meaningful to a certain extent. The usage of this word embedding is the difference between the corpus-based and the knowledge-based. Corpus-based EWC Code RS works relatively similar to knowledge-based in terms of data input, processing, dictionary extraction, sentence vector generation, sentence similarity method, and recommendation generation. The difference is only in the process to determine the word similarity. It uses the cosine coefficient of word vector in word embedding space derived from news corpus instead of WordNet-based word similarity.

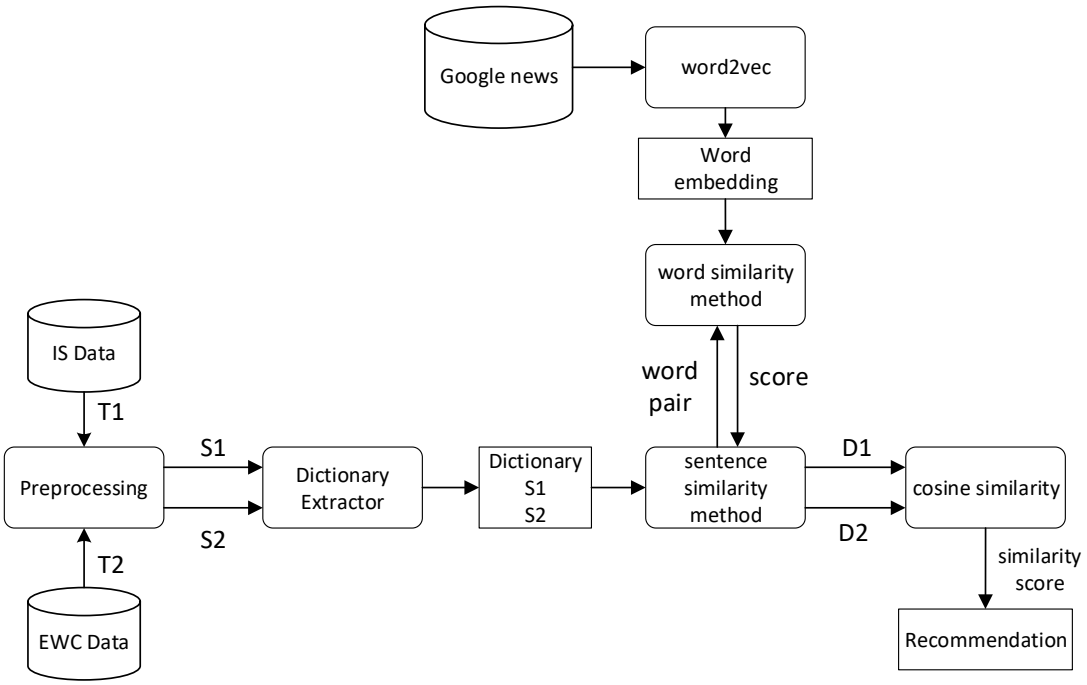


Figure 21. Corpus-based EWC code recommendation system model

For this CB EWC Code RS, we utilize pre-trained word embedding from google, which available in Google Code repository[4]. It is trained using Word2Vec method on Google News corpus comprise 100 billion words. The word embedding model consists of a word vector with 300-dimension long, and it covers 3 million words and phrase. The reason we use this model is two folds. First, corpus resource specific to IS or environmental field doesn't exist. Second, we don't have a computing resource to conduct word2vec training. This is just single research using

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

a personal laptop. Word2vec training on huge corpus requires powerful computing machine to complete in a reasonable time.

4.6. Evaluation Method

The recommender system output the Top-N EWC codes as a recommendation based on the similarity between waste item description with EWC codes description. Therefore, to evaluate the performance, we use evaluation metrics for Top-N recommender system from Desphande and Karypis [74]. The metrics are hit-rate or recall and average reciprocal hit rank (ARHR). We also add precision as an additional metric. Precision is the fraction of recommended items that are relevant to a specific user. Recall is the fraction of relevant recommended items that are retrieved. ARHR is calculated similarly with recall but, the position of where the human-annotated EWC code matches the EWC code recommendation is also being considered. We also use the entire waste item in IS data as a test set.

In our problem context, we evaluate the precision, recall, and ARHR by using equation (14), (15), and (16). For each waste item that has valid human-annotated EWC code, the recommender will output Top-N EWC code. A *hit* happens when the annotated EWC code is present in the Top-N EWC code list. N is the number of recommended EWC codes for each waste item. We use the N value of 5, 10, 15, and 20. *n* is the total number of waste items. A *recommendation* is a situation where the system is able to give a recommendation, and it is counted as 1 even if the system is only able to recommend EWC codes able to recommend codes whose number is below the value of N. In recall, the position where a *hit* happens is not considered, and it is always counted as 1, while in ARHR, if the *hit* happens in position p_i then the *hit* is counted as $\frac{1}{p_i}$.

$$Recall = \frac{Number\ of\ hits}{n} \tag{14}$$

$$Average\ reciprocal\ hit-rank\ (ARHR) = \frac{1}{n}\sum_{i=1}^{h}\frac{1}{p_i} \tag{15}$$

$$Precision = \frac{Number\ of\ hits}{Number\ of\ recommendation} \tag{16}$$

Example:

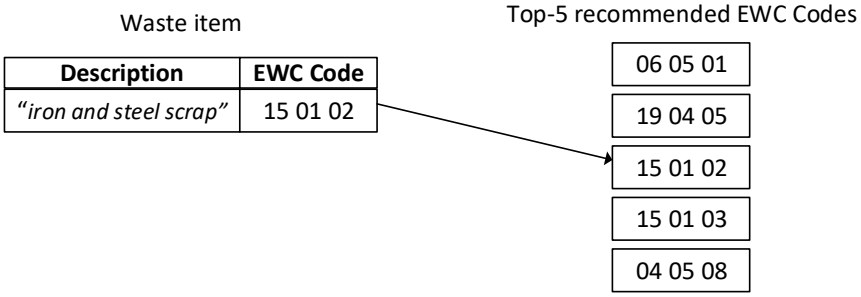


Figure 22. Evaluation method illustration.

For example, let us consider that there are 356 waste items with valid human-annotated EWC code. One of those is a waste item that has human-annotated EWC code of 15 01 02, as illustrated in Figure 22. We assume that N = 5, so the recommender will return Top-5 relevant EWC code based on the sentence similarity score. We can see in Figure 22 that human annotated label code is present in the Top-5 recommended EWC codes list. We called this condition as a *hit*. For all waste items, we count the total number of *hit* that occurred. If the number of *hits* is 125, for example, then the recall of the recommender is calculated using equation (14) as:

- $Recall = \frac{125}{356} = 0.3511$

We want to give higher importance to the *hit* that happened in the higher position of recommendation list since the code in a higher position is more visible to the user. In Figure 22, the *hit* happens in the third position of the list. So, reciprocal hit-rank (RHR) for that waste item is calculated as:

- $RHR = \frac{1}{3} = 0.3333$

To calculate the ARHR, we summarize the value of RHR for each waste item in the dataset and divide it with the total number of waste item using equation (15). If all *hit* happened in the first position of the EWC Code recommendation list, then ARHR value will be the same as recall.

In Figure 22, we can also see the system is able to give 5 EWC codes as recommended items in Top-5 recommendation system. If the system is able to recommend items, we count this as a 1 *recommendation*. If the system is only able to give 3 EWC code in Top-5 recommendation system, we also count it as a 1 *recommendation.* If, for example, the total number of *recommendation* for all waste items is 301 and the total number of *hit* is 125, then the precision is calculated using equation (16) as:

- $Precision = \frac{125}{301} = 0.4153$

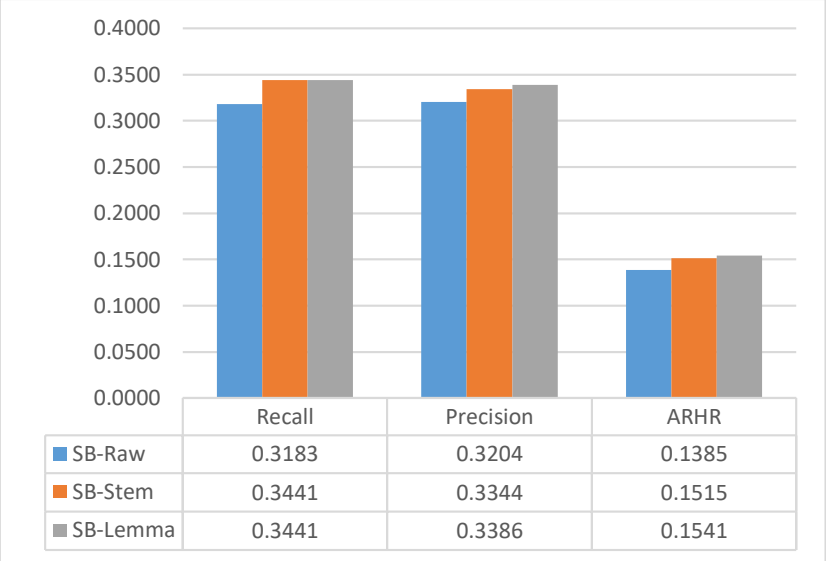
5. Result and discussion

In this section, we will review the effect of parameters change such as stemming/lemmatization and word similarity method to the performance of each Recommender System. After the parameter settings that provide optimal results are known, each Recommender system will apply the setting. Then, the performance of each Recommender system will be compared to find out how the effects of applying semantic awareness to the EWC RS Code.

5.1. String-based EWC Code RS

5.1.1. Effect of stemming and lemmatization

We conduct an experiment to determine the effect of stemming and lemmatization, as described in section 4.3 (data preprocessing) to the quality of the recommendation of string-based EWC Code RS. Figure 23 shows the recall, precision, and ARHR achieved by string-based EWC Code RS using three terms preprocessing step. The label “SB-Raw” refers to the string-based method that doesn’t preprocess the terms contained in the dataset and use the term as it is without any changes, whereas the “SB-Stem” means the terms are stemmed and “SB-Lemma” means the terms are derived into its base form using a lemmatization process before sentence vector generation. We use the value of *N=10* and Top-N recommended EWC codes are retrieved.



	Recall	Precision	ARHR
SB-Raw	0.3183	0.3204	0.1385
SB-Stem	0.3441	0.3344	0.1515
SB-Lemma	0.3441	0.3386	0.1541

Figure 23. Effect of stemming and lemmatization on the quality of string-based EWC Code RS

We can see in Figure 23 that SB-Lemma outperform all other string-based methods that use different preprocessing steps. In recall, SB-Lemma achieved 8 % improvement from SB-Raw. Even though SB-Stem also achieved a similar improvement in Recall, but in Precision and ARHR, it performs worse than SB-Lemma. SB-Lemma could achieve 6 % and 11 % improvement compared to Stem-Raw while SB-Stem only 4 % and 9 % for Precision and ARHR. Considering this fact, we will use lemmatization for other recommendation system methods for the rest of the experiments. Experiment result that uses *N* value other than 10 can be seen in Figure 28.

An explanation why stemming/lemmatization gives better performance is due to the fact that it increases the number of word matching between the vector vocabulary and sentence term in the process of sentence vector generation and TF-IDF weighting. Without a lemmatization or stemming, a slightly different word such as “waste” and “wastes” will be considered as different words even if it is actually the same. In our dataset, lemmatization provides a higher number of word matching than stemming, which leads to better performance for recommender systems. However, lemmatization is computationally more expensive than stemming since it needs to access additional language dictionary to perform the process.

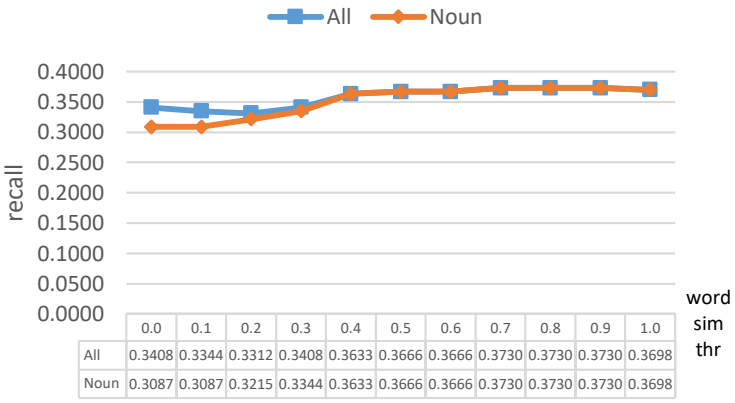
5.2. Knowledge-based EWC Code RS

5.2.1. Effect word type selection

Recall from section 4.5.1.2, we utilize the similarity of the terms that built up the sentences when we calculate short-text/sentence similarity. We query a synonym set (synset) for each term to WordNet, and it will return synset including the word type that a word in a synset belongs to, e.g., noun, verb, adjective, adverb. Our domain problem is about comparing a waste item description and EWC code description where both descriptions mostly are noun phrases. We suspect that term with noun type will have a more dominant role in determining sentence similarity. Therefore, we make two kinds of word type selection that filter words to be processed for sentence vector generation. One only use the term with noun word type while the other use all word types. We perform an experiment to evaluate the effect of this word type selection to

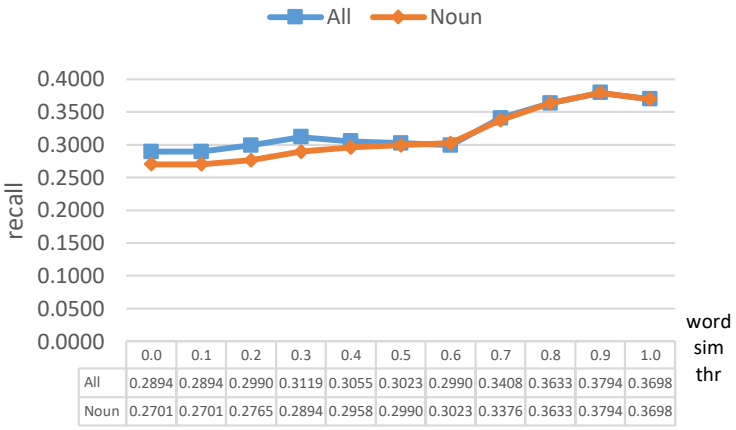
the performance of the EWC Code RS that is specific to the knowledge-based method. The complete experiment result can be seen in Appendix A.

Figure 24 (a), (b), (c) and (c) shows the effect of word type selection to the recall in various word similarity methods, e.g., PATH, WP, LIN, LI as discussed in section 4.5.1.1. We also use a range of word similarity threshold from 0.0 to 1.0, with an increment of 0.1. Value of 1.0 means the terms must be an exact match or a pair of synonym, while 0.0 means the opposite or dissimilar. The label "All" all refers to knowledge-based methods that process all terms available in the dataset while label "Noun" means that we only process term with noun type.



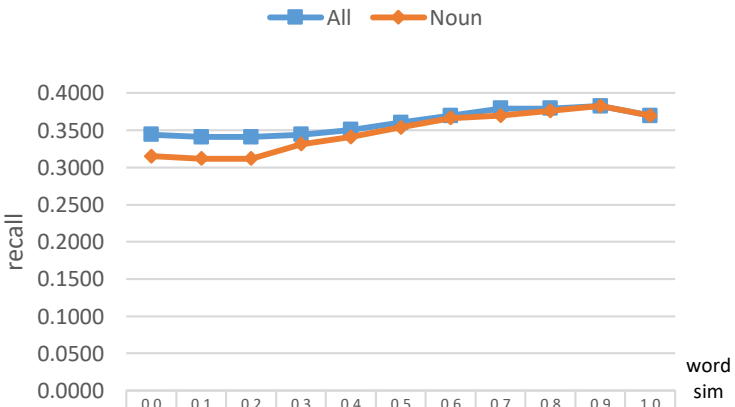
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
All	0.3408	0.3344	0.3312	0.3408	0.3633	0.3666	0.3666	0.3730	0.3730	0.3730	0.3698
Noun	0.3087	0.3087	0.3215	0.3344	0.3633	0.3666	0.3666	0.3730	0.3730	0.3730	0.3698

(a) PATH Word Similarity



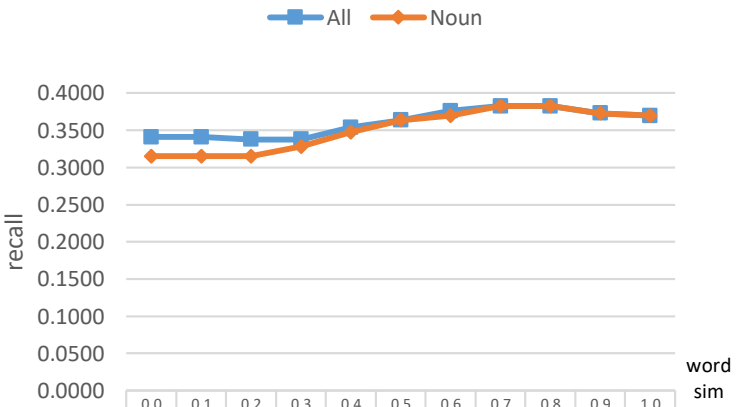
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
All	0.2894	0.2894	0.2990	0.3119	0.3055	0.3023	0.2990	0.3408	0.3633	0.3794	0.3698
Noun	0.2701	0.2701	0.2765	0.2894	0.2958	0.2990	0.3023	0.3376	0.3633	0.3794	0.3698

(b) WP Word Similarity



	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
All	0.3441	0.3408	0.3408	0.3441	0.3505	0.3601	0.3698	0.3794	0.3794	0.3826	0.3698
Noun	0.3151	0.3119	0.3119	0.3312	0.3408	0.3537	0.3666	0.3698	0.3762	0.3826	0.3698

(c) LIN Word Similarity



	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
All	0.3408	0.3408	0.3376	0.3376	0.3537	0.3633	0.3762	0.3826	0.3826	0.3730	0.3698
Noun	0.3151	0.3151	0.3151	0.3280	0.3473	0.3633	0.3698	0.3826	0.3826	0.3730	0.3698

(d) LI Word Similarity

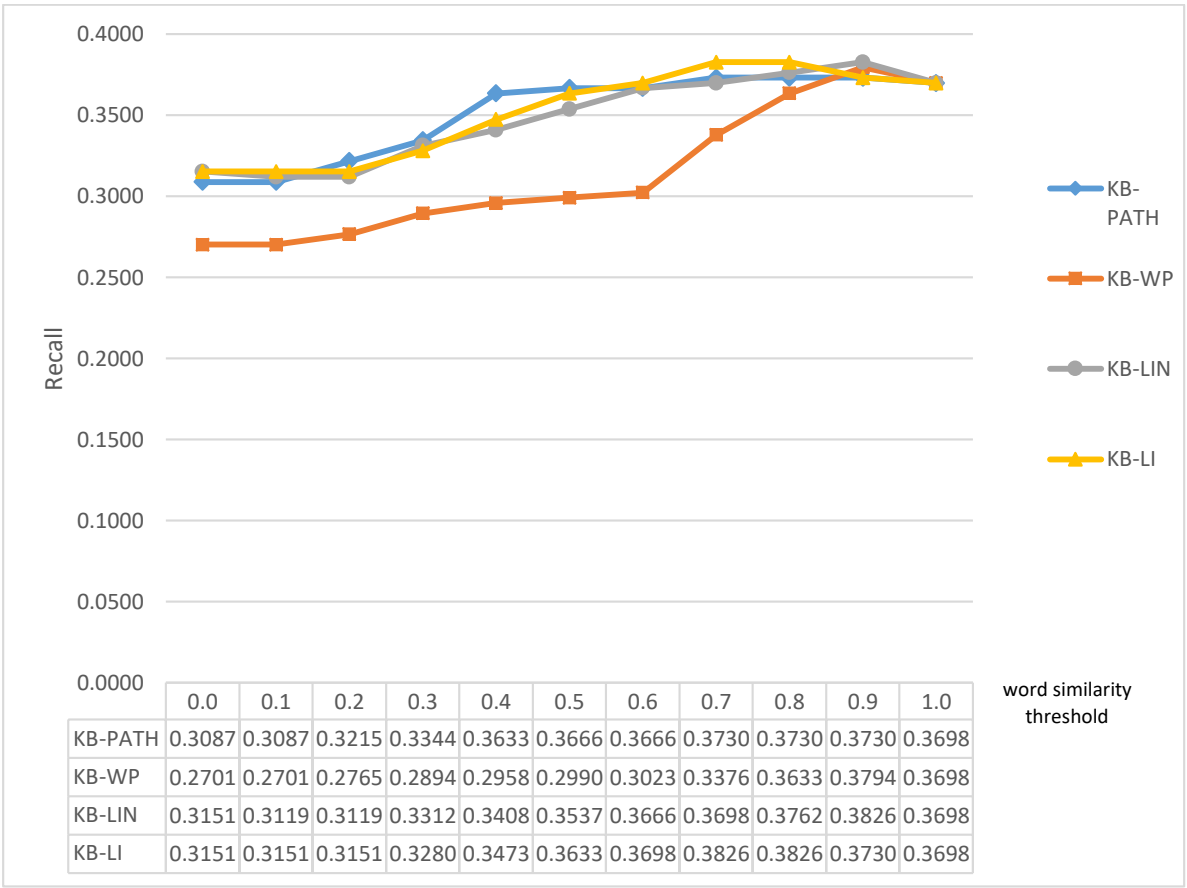
Figure 24. Effect of word type selection on the Recall of knowledge-based EWC Code RS

From the result depicted by Figure 24, we can see knowledge-based methods which process all terms tend to perform better than knowledge-based methods which process only terms with noun type. In threshold value lower than 0.4, the "Noun" can improve the recall up to 10%. However, the maximum recall is achieved in a threshold greater than 0.4, and in that threshold, the recall of "Noun" begin to match up "All". In PATH, the optimal threshold is 0.4 while in WP,

LIN, and LI are 0.5, 0.6, and 0.7 respectively. The use of "Noun" has an additional benefit because the number of the term to be processed is lower than the number of the term in "All" which lead to less computation requirement. Due to this fact, we decide to use the "Noun" type for the remaining knowledge-based method experiment.

5.2.2. Effect of word similarity method and word similarity threshold

As indicated in the previous section, other parameters that affect the performance of knowledge-based method beside word type are word similarity method and threshold. Word similarity method determines the algorithm to measures the similarity between two terms in a semantic network while the threshold determines a boundary value that must be exceeded by the term pair similarity so that the term pair can be processed further to measure sentence similarity. This experiment will evaluate the effect of word similarity method and threshold to the Recall of knowledge-based EWC Code RS. From section 5.2.1, we know that word type "Noun" gives similar performance as "All" word type but has a lower computation; therefore, we set "Noun" as word type selection parameter. We also set *N=10* for Top-N recommender item. In Figure 25, label "KB-PATH" refer to the knowledge-based recommender that uses PATH method as its word similarity whereas "KB-WUP", "KB-LIN", "KB-LI" refer to its own respective word similarity.



	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
KB-PATH	0.3087	0.3087	0.3215	0.3344	0.3633	0.3666	0.3666	0.3730	0.3730	0.3730	0.3698
KB-WP	0.2701	0.2701	0.2765	0.2894	0.2958	0.2990	0.3023	0.3376	0.3633	0.3794	0.3698
KB-LIN	0.3151	0.3119	0.3119	0.3312	0.3408	0.3537	0.3666	0.3698	0.3762	0.3826	0.3698
KB-LI	0.3151	0.3151	0.3151	0.3280	0.3473	0.3633	0.3698	0.3826	0.3826	0.3730	0.3698

word similarity threshold

Figure 25. Effect of word similarity method and word similarity threshold on the recall of knowledge-based EWC Code RS

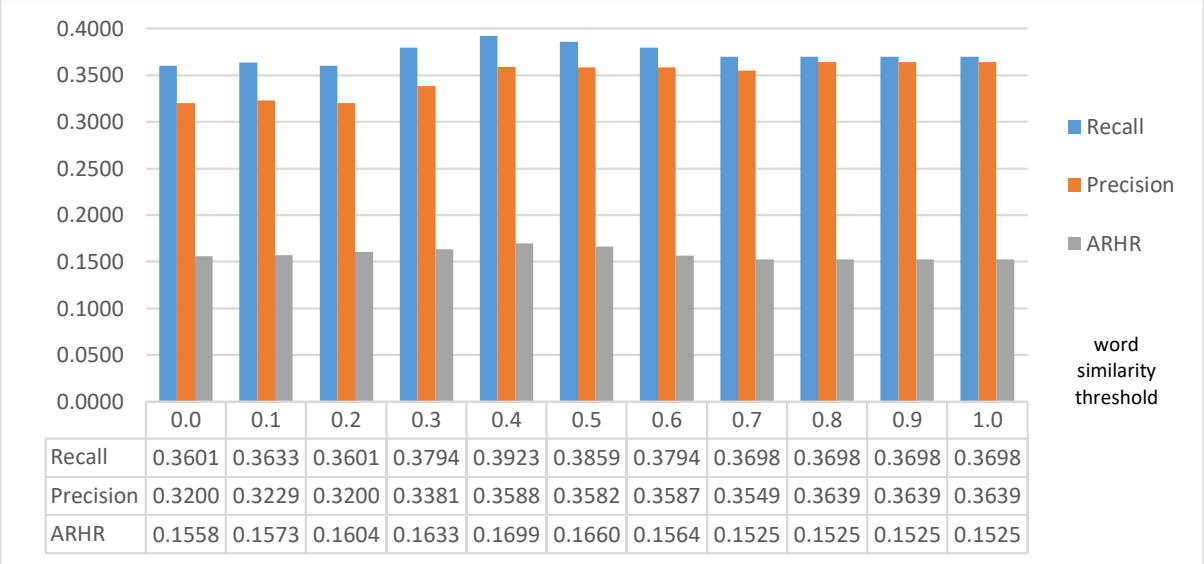
There are several interesting facts that can be observed from the result in Figure 25. (i) For all world similarity methods, threshold value significantly affects recall since different values of threshold lead to a different recall. (ii) In general, KB-WP has the worst performance compared to other methods. (iii) For threshold value below 0.8, KB-WP tends to perform worse compared to other methods, but after 0.8, all the methods have competitive performance. (iv) Increasing the values of the threshold increase the recall up to a certain peak point that is different for each word similarity method. (v) For each word similarity method, the maximum recall is achieved by different threshold values. For KB-PATH, maximum recall is achieved by the threshold value of 0.7, whereas, for KB-WP, KB-LIN, and KB-LI are 0.9, 0.9, and 0.8.

Figure 25 also shows recall comparison between word similarity method in the various threshold. On average, KB-WP is the worst performing method compared with the other three methods so we can use it as a basis of comparison among knowledge-based method. The average improvements that are made over KB-WP by KB-PATH, KB-Lin, and KB-LI are 13%, 12%, 13% respectively. However, if we use the optimal threshold, then all methods achieve slightly different recall.

5.3. Corpus-based EWC Code RS

5.3.1. Effect of word similarity threshold

We perform an experiment to evaluate the effect of word similarity threshold as a parameter that affects the performance of corpus-based EWC Code RS. Figure 26 shows recall, precision, and accuracy achieved by corpus-based methods using various word similarity threshold value from 0.0 to 1.0 with an increment of 0.1. As discussed in section 5.1.1, we use lemmatization as term pre-processing. The complete experiment result can be found in Appendix B.



	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Recall	0.3601	0.3633	0.3601	0.3794	0.3923	0.3859	0.3794	0.3698	0.3698	0.3698	0.3698
Precision	0.3200	0.3229	0.3200	0.3381	0.3588	0.3582	0.3587	0.3549	0.3639	0.3639	0.3639
ARHR	0.1558	0.1573	0.1604	0.1633	0.1699	0.1660	0.1564	0.1525	0.1525	0.1525	0.1525

Figure 26. Effect of word similarity threshold on the quality of corpus-based EWC Code RS

By looking up the result in Figure 26, the maximum value of each evaluation metric is achieved by a different threshold value. The maximum value for the recall is achieved by a threshold value of 0.4 whereas for precision and ARHR is achieved by a threshold value of 0.8 and 0.3 respectively. Since we consider precision as a more important metric, then the other metrics

we decide to use a threshold value of 0.4 for the remaining experiment involving corpus-based method.

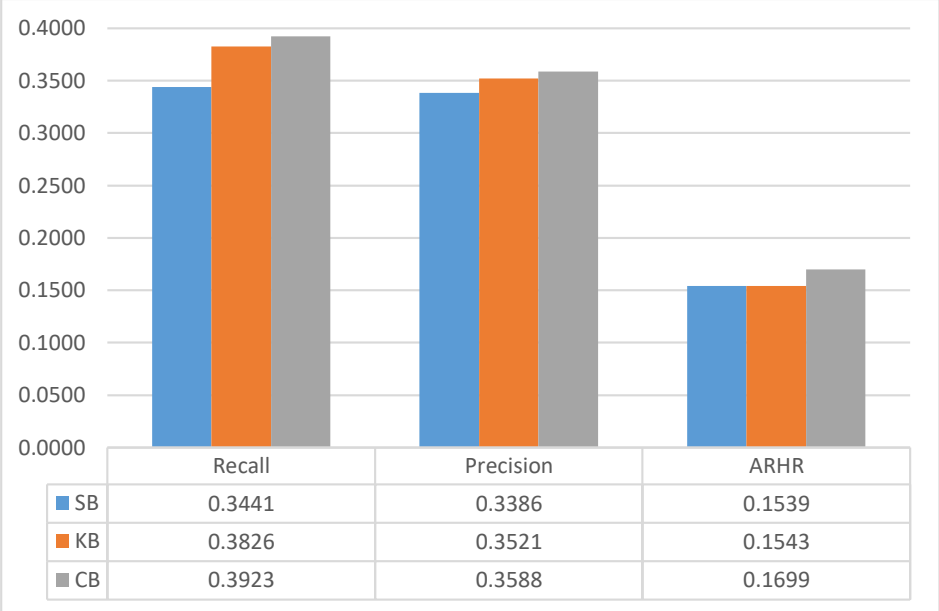
5.4. The comparison of the knowledge-based and corpus-based method with the baseline (string-based method).

From Section 5.1 to 5.3, we have conducted a series of experiments to evaluate the effect of various parameters to the performance of string-based, knowledge-based, and corpus-based EWC Code RS. We are able the determine the optimal parameters that give the best performance for each method from the experiment result we obtained. By using the optimal parameters, we want to compare the performance of each method to evaluate the effect of incorporation of semantic-awareness on EWC Code RS. Both knowledge and corpus-based utilize semantic capabilities, while the string-based doesn't. The optimal parameters used are summarized in Table 10. The label "N/A" means the parameter is not required for the particular method.

Table 10. Optimal parameter settings for EWC Code RS

Method	N	Word similarity threshold	Word similarity method	Pre-processing	Word type
String-based	10	N/A	N/A	lemmatization	N/A
Knowledge-based	10	0.8	Li	lemmatization	Noun
Corpus-based	10	0.4	Word2Vec	lemmatization	N/A

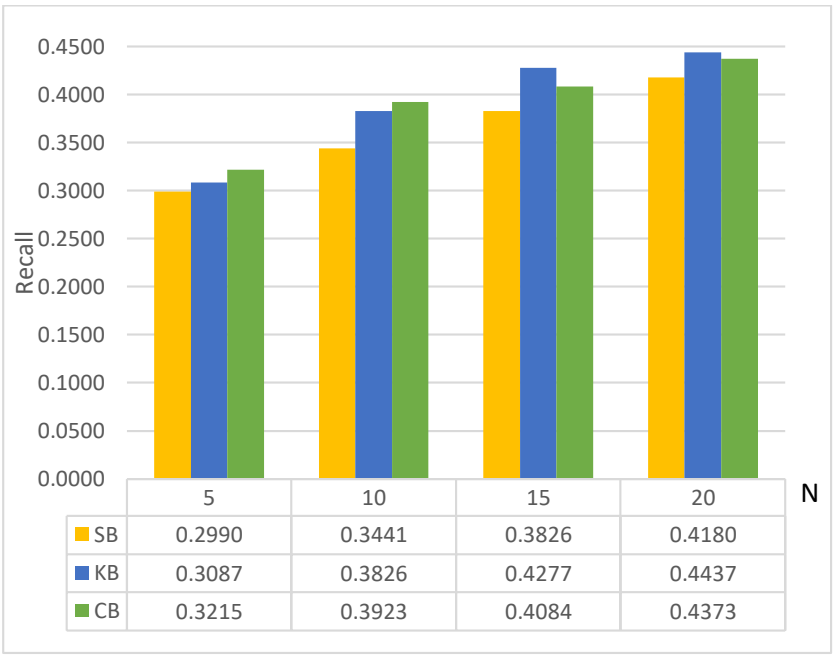
The result of the experiment can be seen in Figure 27. "SB" refers to string-based method while "KB" refers to knowledge-based and "CB" to corpus-based. The result in the figure shows that the corpus-based is the best performing method in all evaluation metrics, including recall, precision, and ARHR. The second best method is knowledge-based. Even though it performs relatively similar to SB in term of ARHR, but in recall and precision, it is significantly better. If we measure the extent of improvement against the baseline (SB) in the recall, precision, and ARHR, KB made improvement of 11%, 4%, and 0.3%. CB achieve even better performance with 14%, 6% and 10% of improvement.



	Recall	Precision	ARHR
SB	0.3441	0.3386	0.1539
KB	0.3826	0.3521	0.1543
CB	0.3923	0.3588	0.1699

Figure 27. Effect of incorporation of semantic awareness on the quality of EWC Code RS

All previous experiments have been conducted by using the N value of 10. We also want to know how the recommenders perform in different values of N. Figure 28 shows the recall achieve by the string-based (SB), knowledge-based (KB) and corpus-based (CB) method for N value of 5, 10, 15 and 20. We limit N value to 20 because we think that by using N value more than 20, then the recommended EWC codes result will be too cluttering the screen when it is displayed to the user. Each recommender method using the optimal parameter, as shown in Table 10.



	5	10	15	20
SB	0.2990	0.3441	0.3826	0.4180
KB	0.3087	0.3826	0.4277	0.4437
CB	0.3215	0.3923	0.4084	0.4373

Figure 28. Recall of the EWC Code Recommender in various values of N

Figure 28 shows several aspects that can be observed. (i) Along with the increase in the value of N, the recall value also increases. For every method, the highest recall is achieved by the

maximum N value (20). This is logical because if the number of recommended EWC codes increases, then the possibility of user-labeled EWC code to be present on the recommendation list will also increase. (ii) In every N value, both knowledge-based and corpus-based outperform string-based methods. However, the improvement is varying in every N value. The maximum improvement is achieved in N value of 10, where SB and CB made an improvement of 11% and 14% compared to SB. Lower or higher N value leads to lower improvement. (iii) For N values below 15, CB has a better performance than KB. However, KB outperformed CB for N value of 15 and above. This fact indicates that KB is not absolutely superior to CB. Nonetheless, the incorporation of semantic-awareness in both methods makes them better than SB.

6. Limitation

There are two limitations to this research. First, we use only four WordNet-based word similarity methods, which are PATH, WP, LIN, and LI. The reason is that they return similarity score in a range 0 to 1 where 0 means dissimilar and 1 otherwise. We can find other WordNet-based methods in the literature such technique from Leacock and Chodorow [29], Resnik [31], Jiang and Conrath [33], but they return similarity score from 0 to infinity not 0 to 1. Normalization strategy needs to be applied. Second, we use general-purpose knowledge and corpus resources such as WordNet and Google news corpus. This is because there is no specific domain corpus and knowledge resource for IS field to the best of our knowledge.

7. Future work

We believe that the result can be improved even further by providing more domain-related knowledge and corpus resource. In this research, we just utilize WordNet and Google news as the resource for knowledge-based and corpus-based method. WordNet is a generic language semantic network while Google News is news corpus from various domain such as politics, science, economy. Both are just generic source that might not contain much specific waste or environmental term. Thus, it only gives decent coverage of the term present in the dataset since the dataset contains a lot of technical terms. The terms give a significant role in the calculation of sentence/short text similarity as the core of the RS since it relies on terms similarity. If the term does not exist in the knowledge or corpus resource, then the term will be ignored. By developing knowledge and corpus resource that closely related to the waste or environmental domain problem, we can expect the term coverage to increase, which in turn affect the sentence similarity positively. In more established domain such as medical, domain-specific knowledge and corpus resource is already exist. They have SNOMED-CT and MESH [65] as their resource.

To improve performance, the use of level 1 and level 2 EWC code descriptions can also be considered in the future. In this research, we only use level 3 of the EWC Code description. In fact, there is a possibility that the waste item description cannot be related to level 3 but with a more general description at level 2 and level 1. Another thing that can be investigated is the use of a company profile if it is available. It is possible that the types of waste items are not clearly represented in the description. If we know that the company is operating in a particular field, the EWC code relating to the field can be given higher priority in the recommendation process.

Another aspect that might be beneficial for research in this field is providing more objective test data. In this research, most of the waste item data is annotated by a single user only. We assume the

data is labeled by correct EWC code even though the correctness is still debatable. The user might not be aware that there is actually more appropriate EWC code for the particular waste item. When the RS recommends Top N EWC code, the user labeled EWC code might not present in the Top-N list even though the correct label is indeed on the list. This will mislead the performance measurement of the recommender system. Thus, it will be beneficial to have several user label EWC code for the same waste item and choose the majority label as the more objective EWC code.

8. Conclusion

We designed SB, KB, and CB EWC Code Recommender method that can recommend the EWC Codes for a waste item by comparing the description of the waste item with the description of the EWC codes. The Top-N most similar codes will be returned by the system as recommendations. In developing this method, we formulated the Research Question and Sub Question SQ as guidelines in developing the methods. By answering RQ and SQ, there are some important aspects that we can conclude from this research as follows.

SQ1: What recommender system method is suitable with the conditions where the user interest is difficult to obtain due to the limited information of user-item interaction?

We conduct a Literature Study on recommender system method, which result can be seen in section 3.1. There are mainly two types of RS, which are collaborative filtering (CF) and content-based (CB). CF recommend an item that was also like by other users that are similar to the active user. CB makes prediction of a recommended item by determining the content or attribute of the item that was liked by the active user and then searched the similar item as recommendation. Both methods are relying on the history of interaction between users and items which are not available in our domain problem. However, there are special case of CB called Knowledge-based recommender which also examine the item attribute. This type of RS matches the item attribute with the knowledge of user interest that is extracted from user input. It is not based on the history of the item liked in the past. In our domain problem, we have very limited history and the user interest is keep changing depending on the waste description he input. This characteristic matches with the property of knowledge-based RS; therefore, we decided that Knowledge-based RS is the most suitable method for our case. The item attribute is extracted from EWC code descriptions while user interest is determined from waste item description that the user input as free text. Then, the recommendation of EWC code is made by comparing these two aspects.

SQ2: What short text similarity measurement method that is available in the literature?

The core of EWC Code RS is the comparison between a waste item description and EWC code description. Both are in the form of short text with a length of fewer than 20 words. Therefore, we conduct a Systematic Literature Review of the Short Text Similarity (STS) measurement method, as described in section 3.2. There are 25 methods that can be categorized into three types. The first type is a string-based method that could compare lexically similar short text. The second and third are knowledge-based and corpus-base. There are also hybrid methods that combine several methods. Knowledge-based rely on a network of concept to determine word-to-word similarity while corpus-based based on relation of word in a context derived from a corpus. We use this information we obtained from SLR as a basis to design EWC Code RS.

SQ3: What is the effect of incorporation of semantic-aware short text similarity measurement method to the accuracy of EWC RS?

The result of the experiment in section 5.4 shows that by adding semantic-awareness, the recall, precision, and ARHR of the recommender will increase compared to the baseline. The knowledge-based improve the metrics of recall, precision, and ARHR by 11%, 12%, 6% whereas the corpus-based improve the metrics by 14%, 7%, 5% in Top-10 recommendation.

SQ4: How does short text preprocessing (e.g., stemming, lemmatization) affect the quality of EWC RS?

Based on the experiment result described in section 5.1.1, we found that text input with stemming and lemmatization increase precision, recall, and ARHR of string-based EWC Code RS compared to raw text without any pre-processing. We decided to also use lemmatization on knowledge-based and corpus-based EWC Code RS. One of the reason is that WordNet which is used in knowledge-based EWC RS also stores words in its basic form without inflection. Therefore, lemmatization will increase the matching word-pair between text input and WordNet, which lead to a performance increase.

SQ5: How does the word similarity method affect the quality of EWC RS?

We incorporate word similarity metric to add semantic awareness capability to knowledge-bases and corpus-based EWC Code RS. The knowledge-based use four different WordNet-based word similarity metrics, which are Path, WP, Lin, and Li. The corpus-based use word similarity method derived from word2vec trained on Google News corpus. Experiment result in section 4.6 shows the method using Li word similarity is the best performing method for knowledge-based. However, this method is outperformed by the method using word vector from word2vec according to the experiment results in section 5.4

Main RQ: Given a waste product description in IS marketplace, can we accurately recommend EWC code that the product belongs to?

We propose three different methods that can recommend EWC codes using only waste item description without any historical data interaction between users and EWC code. The methods are SB, KB, and CB. SB uses exact match string matching while KB and CB are able to identify semantically similar words. All methods work by comparing the waste item description and each entry of EWC code description and look for Top-N most similar text list. The EWC codes of this Top-N description are returned as recommended EWC code. The text comparison is conducted by transforming both descriptions into its sentence, or semantic vectors form using the joint word set from both description as vector vocabulary. The text similarity is determined by calculating the cosine coefficient between these two sentence vectors. The difference between the three methods lies determining word-to-word similarity to generates the sentence vector element. In SB, the word pair must be exact match while in KB uses WordNet-based word similarity, and CB uses word vector form word2vec model. We evaluate the performance of the method in the Top-10 EWC Code recommender setting. The result in section 5.4 shows the SB method could achieve recall, precision, and ARHR by 34.4%, 33.9%, and 15.4%. The KB which utilize semantic-awareness could achieve better performance by 38.3%, 35.2%, and 15.4%. The CB perform even higher by 39.2%, 35.9%, and 16.7%.

Appendix A

The test result of the Top-10 Knowledge-based EWC Code Recommender System.

word sim method	pre-process	word type	word sim threshold	no_rec	precision	recall	ARHR
PATH	raw	all	0	351	0.2650	0.2990	0.1322
PATH	raw	all	0.1	351	0.2650	0.2990	0.1319
PATH	raw	all	0.2	351	0.2735	0.3087	0.1349
PATH	raw	all	0.3	351	0.2792	0.3151	0.1383
PATH	raw	all	0.4	351	0.2877	0.3248	0.1426
PATH	raw	all	0.5	343	0.3120	0.3441	0.1483
PATH	raw	all	0.6	335	0.3224	0.3473	0.1473
PATH	raw	all	0.7	330	0.3303	0.3505	0.1498
PATH	raw	all	0.8	328	0.3323	0.3505	0.1505
PATH	raw	all	0.9	328	0.3323	0.3505	0.1505
PATH	raw	all	1	309	0.3204	0.3183	0.1380
PATH	raw	noun	0	351	0.2536	0.2862	0.1251
PATH	raw	noun	0.1	351	0.2536	0.2862	0.1247
PATH	raw	noun	0.2	351	0.2678	0.3023	0.1318
PATH	raw	noun	0.3	351	0.2821	0.3183	0.1353
PATH	raw	noun	0.4	351	0.2906	0.3280	0.1404
PATH	raw	noun	0.5	342	0.3129	0.3441	0.1456
PATH	raw	noun	0.6	333	0.3243	0.3473	0.1449
PATH	raw	noun	0.7	327	0.3333	0.3505	0.1473
PATH	raw	noun	0.8	325	0.3354	0.3505	0.1480
PATH	raw	noun	0.9	325	0.3354	0.3505	0.1480
PATH	raw	noun	1	309	0.3204	0.3183	0.1380
PATH	stem	all	0	351	0.2279	0.2572	0.1187
PATH	stem	all	0.1	351	0.2279	0.2572	0.1187
PATH	stem	all	0.2	351	0.2251	0.2540	0.1172
PATH	stem	all	0.3	351	0.2194	0.2476	0.1122
PATH	stem	all	0.4	351	0.2279	0.2572	0.1141
PATH	stem	all	0.5	350	0.2714	0.3055	0.1271
PATH	stem	all	0.6	347	0.2997	0.3344	0.1362
PATH	stem	all	0.7	339	0.3274	0.3569	0.1477
PATH	stem	all	0.8	330	0.3485	0.3698	0.1483
PATH	stem	all	0.9	330	0.3485	0.3698	0.1515
PATH	stem	all	1	320	0.3594	0.3698	0.1502
PATH	stem	noun	0	351	0.2279	0.2572	0.1193
PATH	stem	noun	0.1	351	0.2279	0.2572	0.1193
PATH	stem	noun	0.2	351	0.2251	0.2540	0.1178
PATH	stem	noun	0.3	351	0.2165	0.2444	0.1118
PATH	stem	noun	0.4	351	0.2279	0.2572	0.1141
PATH	stem	noun	0.5	350	0.2714	0.3055	0.1271
PATH	stem	noun	0.6	347	0.2968	0.3312	0.1358
PATH	stem	noun	0.7	338	0.3284	0.3569	0.1476
PATH	stem	noun	0.8	327	0.3517	0.3698	0.1482
PATH	stem	noun	0.9	326	0.3528	0.3698	0.1515
PATH	stem	noun	1	320	0.3594	0.3698	0.1502
PATH	lemma	all	0	351	0.3020	0.3408	0.1437
PATH	lemma	all	0.1	351	0.2963	0.3344	0.1465
PATH	lemma	all	0.2	351	0.2934	0.3312	0.1463
PATH	lemma	all	0.3	351	0.3020	0.3408	0.1478
PATH	lemma	all	0.4	351	0.3219	0.3633	0.1501
PATH	lemma	all	0.5	343	0.3324	0.3666	0.1559
PATH	lemma	all	0.6	337	0.3383	0.3666	0.1538
PATH	lemma	all	0.7	330	0.3515	0.3730	0.1549
PATH	lemma	all	0.8	328	0.3537	0.3730	0.1556
PATH	lemma	all	0.9	328	0.3537	0.3730	0.1556
PATH	lemma	all	1	316	0.3639	0.3698	0.1525
PATH	lemma	noun	0	351	0.2735	0.3087	0.1354
PATH	lemma	noun	0.1	351	0.2735	0.3087	0.1386
PATH	lemma	noun	0.2	351	0.2849	0.3215	0.1423
PATH	lemma	noun	0.3	351	0.2963	0.3344	0.1420

word sim method	pre-process	word type	word sim threshold	no_rec	precision	recall	ARHR
PATH	lemma	noun	0.4	351	0.3219	0.3633	0.1477
PATH	lemma	noun	0.5	342	0.3333	0.3666	0.1536
PATH	lemma	noun	0.6	335	0.3403	0.3666	0.1515
PATH	lemma	noun	0.7	327	0.3547	0.3730	0.1527
PATH	lemma	noun	0.8	325	0.3569	0.3730	0.1534
PATH	lemma	noun	0.9	325	0.3569	0.3730	0.1534
PATH	lemma	noun	1	316	0.3639	0.3698	0.1525
WP	raw	all	0	351	0.2165	0.2444	0.1119
WP	raw	all	0.1	351	0.2165	0.2444	0.1119
WP	raw	all	0.2	351	0.2165	0.2444	0.1114
WP	raw	all	0.3	351	0.2251	0.2540	0.1135
WP	raw	all	0.4	351	0.2251	0.2540	0.1165
WP	raw	all	0.5	347	0.2305	0.2572	0.1204
WP	raw	all	0.6	347	0.2421	0.2701	0.1247
WP	raw	all	0.7	347	0.2622	0.2926	0.1342
WP	raw	all	0.8	347	0.3112	0.3473	0.1427
WP	raw	all	0.9	345	0.3130	0.3473	0.1475
WP	raw	all	1	309	0.3204	0.3183	0.1380
WP	raw	noun	0	351	0.2165	0.2444	0.1107
WP	raw	noun	0.1	351	0.2165	0.2444	0.1107
WP	raw	noun	0.2	351	0.2165	0.2444	0.1103
WP	raw	noun	0.3	351	0.2222	0.2508	0.1110
WP	raw	noun	0.4	351	0.2251	0.2540	0.1140
WP	raw	noun	0.5	347	0.2334	0.2605	0.1197
WP	raw	noun	0.6	347	0.2421	0.2701	0.1235
WP	raw	noun	0.7	347	0.2651	0.2958	0.1331
WP	raw	noun	0.8	347	0.3112	0.3473	0.1407
WP	raw	noun	0.9	345	0.3130	0.3473	0.1452
WP	raw	noun	1	309	0.3204	0.3183	0.1380
WP	stem	all	0	351	0.2165	0.2444	0.1195
WP	stem	all	0.1	351	0.2165	0.2444	0.1195
WP	stem	all	0.2	351	0.2165	0.2444	0.1174
WP	stem	all	0.3	351	0.2222	0.2508	0.1169
WP	stem	all	0.4	351	0.2165	0.2444	0.1110
WP	stem	all	0.5	351	0.2365	0.2669	0.1173
WP	stem	all	0.6	351	0.2650	0.2990	0.1261
WP	stem	all	0.7	345	0.2986	0.3312	0.1340
WP	stem	all	0.8	341	0.3226	0.3537	0.1431
WP	stem	all	0.9	337	0.3442	0.3730	0.1487
WP	stem	all	1	320	0.3594	0.3698	0.1502
WP	stem	noun	0	351	0.2108	0.2379	0.1162
WP	stem	noun	0.1	351	0.2108	0.2379	0.1162
WP	stem	noun	0.2	351	0.2080	0.2347	0.1162
WP	stem	noun	0.3	351	0.2222	0.2508	0.1164
WP	stem	noun	0.4	351	0.2137	0.2412	0.1108
WP	stem	noun	0.5	351	0.2336	0.2637	0.1172
WP	stem	noun	0.6	351	0.2764	0.3119	0.1270
WP	stem	noun	0.7	345	0.2986	0.3312	0.1337
WP	stem	noun	0.8	341	0.3284	0.3601	0.1437
WP	stem	noun	0.9	337	0.3442	0.3730	0.1489
WP	stem	noun	1	320	0.3594	0.3698	0.1502
WP	lemma	all	0	351	0.2564	0.2894	0.1287
WP	lemma	all	0.1	351	0.2564	0.2894	0.1287
WP	lemma	all	0.2	351	0.2650	0.2990	0.1293
WP	lemma	all	0.3	351	0.2764	0.3119	0.1325
WP	lemma	all	0.4	351	0.2707	0.3055	0.1328
WP	lemma	all	0.5	347	0.2709	0.3023	0.1345
WP	lemma	all	0.6	347	0.2680	0.2990	0.1406
WP	lemma	all	0.7	347	0.3055	0.3408	0.1439
WP	lemma	all	0.8	347	0.3256	0.3633	0.1514
WP	lemma	all	0.9	345	0.3420	0.3794	0.1527
WP	lemma	all	1	316	0.3639	0.3698	0.1525
WP	lemma	noun	0	351	0.2393	0.2701	0.1245

word sim method	pre-process	word type	word sim threshold	no_rec	precision	recall	ARHR
WP	lemma	noun	0.1	351	0.2393	0.2701	0.1245
WP	lemma	noun	0.2	351	0.2450	0.2765	0.1248
WP	lemma	noun	0.3	351	0.2564	0.2894	0.1276
WP	lemma	noun	0.4	351	0.2621	0.2958	0.1310
WP	lemma	noun	0.5	347	0.2680	0.2990	0.1343
WP	lemma	noun	0.6	347	0.2709	0.3023	0.1385
WP	lemma	noun	0.7	347	0.3026	0.3376	0.1412
WP	lemma	noun	0.8	347	0.3256	0.3633	0.1492
WP	lemma	noun	0.9	345	0.3420	0.3794	0.1508
WP	lemma	noun	1	316	0.3639	0.3698	0.1525
LIN	raw	all	0	351	0.2422	0.2733	0.1270
LIN	raw	all	0.1	351	0.2422	0.2733	0.1282
LIN	raw	all	0.2	351	0.2422	0.2733	0.1277
LIN	raw	all	0.3	351	0.2479	0.2797	0.1281
LIN	raw	all	0.4	351	0.2564	0.2894	0.1321
LIN	raw	all	0.5	346	0.2919	0.3248	0.1377
LIN	raw	all	0.6	345	0.3043	0.3376	0.1423
LIN	raw	all	0.7	345	0.3217	0.3569	0.1540
LIN	raw	all	0.8	343	0.3353	0.3698	0.1477
LIN	raw	all	0.9	342	0.3216	0.3537	0.1502
LIN	raw	all	1	309	0.3204	0.3183	0.1380
LIN	raw	noun	0	351	0.2450	0.2765	0.1286
LIN	raw	noun	0.1	351	0.2450	0.2765	0.1297
LIN	raw	noun	0.2	351	0.2450	0.2765	0.1280
LIN	raw	noun	0.3	351	0.2564	0.2894	0.1291
LIN	raw	noun	0.4	351	0.2821	0.3183	0.1359
LIN	raw	noun	0.5	346	0.3064	0.3408	0.1425
LIN	raw	noun	0.6	345	0.3246	0.3601	0.1452
LIN	raw	noun	0.7	344	0.3227	0.3569	0.1467
LIN	raw	noun	0.8	341	0.3255	0.3569	0.1458
LIN	raw	noun	0.9	338	0.3254	0.3537	0.1477
LIN	raw	noun	1	309	0.3204	0.3183	0.1380
LIN	stem	all	0	351	0.2165	0.2444	0.1167
LIN	stem	all	0.1	351	0.2165	0.2444	0.1167
LIN	stem	all	0.2	351	0.2165	0.2444	0.1170
LIN	stem	all	0.3	351	0.2251	0.2540	0.1183
LIN	stem	all	0.4	351	0.2251	0.2540	0.1151
LIN	stem	all	0.5	351	0.2593	0.2926	0.1247
LIN	stem	all	0.6	351	0.2877	0.3248	0.1341
LIN	stem	all	0.7	345	0.3188	0.3537	0.1465
LIN	stem	all	0.8	338	0.3284	0.3569	0.1440
LIN	stem	all	0.9	338	0.3462	0.3762	0.1516
LIN	stem	all	1	320	0.3594	0.3698	0.1502
LIN	stem	noun	0	351	0.2165	0.2444	0.1163
LIN	stem	noun	0.1	351	0.2165	0.2444	0.1163
LIN	stem	noun	0.2	351	0.2165	0.2444	0.1162
LIN	stem	noun	0.3	351	0.2251	0.2540	0.1162
LIN	stem	noun	0.4	351	0.2251	0.2540	0.1145
LIN	stem	noun	0.5	351	0.2536	0.2862	0.1244
LIN	stem	noun	0.6	351	0.2906	0.3280	0.1339
LIN	stem	noun	0.7	345	0.3217	0.3569	0.1468
LIN	stem	noun	0.8	338	0.3314	0.3601	0.1443
LIN	stem	noun	0.9	337	0.3531	0.3826	0.1522
LIN	stem	noun	1	320	0.3594	0.3698	0.1502
LIN	lemma	all	0	351	0.3048	0.3441	0.1434
LIN	lemma	all	0.1	351	0.3020	0.3408	0.1436
LIN	lemma	all	0.2	351	0.3020	0.3408	0.1432
LIN	lemma	all	0.3	351	0.3048	0.3441	0.1450
LIN	lemma	all	0.4	351	0.3105	0.3505	0.1499
LIN	lemma	all	0.5	346	0.3237	0.3601	0.1517
LIN	lemma	all	0.6	345	0.3333	0.3698	0.1567
LIN	lemma	all	0.7	345	0.3420	0.3794	0.1616
LIN	lemma	all	0.8	343	0.3440	0.3794	0.1556

word sim method	pre-process	word type	word sim threshold	no_rec	precision	recall	ARHR
LIN	lemma	all	0.9	342	0.3480	0.3826	0.1566
LIN	lemma	all	1	316	0.3639	0.3698	0.1525
LIN	lemma	noun	0	351	0.2792	0.3151	0.1370
LIN	lemma	noun	0.1	351	0.2764	0.3119	0.1388
LIN	lemma	noun	0.2	351	0.2764	0.3119	0.1383
LIN	lemma	noun	0.3	351	0.2934	0.3312	0.1409
LIN	lemma	noun	0.4	351	0.3020	0.3408	0.1464
LIN	lemma	noun	0.5	346	0.3179	0.3537	0.1487
LIN	lemma	noun	0.6	345	0.3304	0.3666	0.1475
LIN	lemma	noun	0.7	344	0.3343	0.3698	0.1516
LIN	lemma	noun	0.8	341	0.3431	0.3762	0.1528
LIN	lemma	noun	0.9	338	0.3521	0.3826	0.1543
LIN	lemma	noun	1	316	0.3639	0.3698	0.1525
LI	raw	all	0	351	0.2479	0.2797	0.1272
LI	raw	all	0.1	351	0.2479	0.2797	0.1274
LI	raw	all	0.2	351	0.2450	0.2765	0.1279
LI	raw	all	0.3	351	0.2536	0.2862	0.1290
LI	raw	all	0.4	351	0.2735	0.3087	0.1342
LI	raw	all	0.5	347	0.2911	0.3248	0.1446
LI	raw	all	0.6	347	0.3141	0.3505	0.1477
LI	raw	all	0.7	345	0.3130	0.3473	0.1449
LI	raw	all	0.8	342	0.3187	0.3505	0.1459
LI	raw	all	0.9	325	0.3354	0.3505	0.1479
LI	raw	all	1	309	0.3204	0.3183	0.1380
LI	raw	noun	0	351	0.2507	0.2830	0.1272
LI	raw	noun	0.1	351	0.2507	0.2830	0.1271
LI	raw	noun	0.2	351	0.2450	0.2765	0.1247
LI	raw	noun	0.3	351	0.2536	0.2862	0.1306
LI	raw	noun	0.4	351	0.2792	0.3151	0.1353
LI	raw	noun	0.5	347	0.2939	0.3280	0.1428
LI	raw	noun	0.6	347	0.3141	0.3505	0.1455
LI	raw	noun	0.7	342	0.3187	0.3505	0.1429
LI	raw	noun	0.8	342	0.3187	0.3505	0.1436
LI	raw	noun	0.9	325	0.3354	0.3505	0.1462
LI	raw	noun	1	309	0.3204	0.3183	0.1380
LI	stem	all	0	351	0.2222	0.2508	0.1186
LI	stem	all	0.1	351	0.2222	0.2508	0.1186
LI	stem	all	0.2	351	0.2194	0.2476	0.1186
LI	stem	all	0.3	351	0.2251	0.2540	0.1181
LI	stem	all	0.4	351	0.2165	0.2444	0.1128
LI	stem	all	0.5	351	0.2678	0.3023	0.1283
LI	stem	all	0.6	351	0.2991	0.3376	0.1379
LI	stem	all	0.7	345	0.3304	0.3666	0.1484
LI	stem	all	0.8	336	0.3482	0.3762	0.1490
LI	stem	all	0.9	327	0.3517	0.3698	0.1515
LI	stem	all	1	320	0.3594	0.3698	0.1502
LI	stem	noun	0	351	0.2194	0.2476	0.1180
LI	stem	noun	0.1	351	0.2194	0.2476	0.1180
LI	stem	noun	0.2	351	0.2165	0.2444	0.1179
LI	stem	noun	0.3	351	0.2222	0.2508	0.1172
LI	stem	noun	0.4	351	0.2194	0.2476	0.1130
LI	stem	noun	0.5	351	0.2678	0.3023	0.1281
LI	stem	noun	0.6	351	0.2991	0.3376	0.1379
LI	stem	noun	0.7	345	0.3304	0.3666	0.1485
LI	stem	noun	0.8	336	0.3482	0.3762	0.1491
LI	stem	noun	0.9	326	0.3528	0.3698	0.1515
LI	stem	noun	1	320	0.3594	0.3698	0.1502
LI	lemma	all	0	351	0.3020	0.3408	0.1393
LI	lemma	all	0.1	351	0.3020	0.3408	0.1429
LI	lemma	all	0.2	351	0.2991	0.3376	0.1435
LI	lemma	all	0.3	351	0.2991	0.3376	0.1435
LI	lemma	all	0.4	351	0.3134	0.3537	0.1514
LI	lemma	all	0.5	347	0.3256	0.3633	0.1561

word sim method	pre-process	word type	word sim threshold	no_rec	precision	recall	ARHR
LI	lemma	all	0.6	347	0.3372	0.3762	0.1546
LI	lemma	all	0.7	345	0.3449	0.3826	0.1531
LI	lemma	all	0.8	342	0.3480	0.3826	0.1538
LI	lemma	all	0.9	325	0.3569	0.3730	0.1551
LI	lemma	all	1	316	0.3639	0.3698	0.1525
LI	lemma	noun	0	351	0.2792	0.3151	0.1355
LI	lemma	noun	0.1	351	0.2792	0.3151	0.1388
LI	lemma	noun	0.2	351	0.2792	0.3151	0.1403
LI	lemma	noun	0.3	351	0.2906	0.3280	0.1392
LI	lemma	noun	0.4	351	0.3077	0.3473	0.1492
LI	lemma	noun	0.5	347	0.3256	0.3633	0.1537
LI	lemma	noun	0.6	347	0.3314	0.3698	0.1520
LI	lemma	noun	0.7	342	0.3480	0.3826	0.1512
LI	lemma	noun	0.8	342	0.3480	0.3826	0.1519
LI	lemma	noun	0.9	325	0.3569	0.3730	0.1534
LI	lemma	noun	1	316	0.3639	0.3698	0.1525

Appendix B

The test result of the Top-10 Corpus-based EWC Code Recommender System.

word sim method	pre-process	word type	word sim threshold	no_rec	precision	recall	ARHR
Word2Vec	raw	all	0	350	0.2686	0.3023	0.1340
Word2Vec	raw	all	0.1	350	0.2686	0.3023	0.1331
Word2Vec	raw	all	0.2	350	0.2686	0.3023	0.1366
Word2Vec	raw	all	0.3	349	0.2722	0.3055	0.1402
Word2Vec	raw	all	0.4	343	0.3032	0.3344	0.1435
Word2Vec	raw	all	0.5	334	0.3204	0.3441	0.1434
Word2Vec	raw	all	0.6	326	0.3221	0.3376	0.1411
Word2Vec	raw	all	0.7	321	0.3146	0.3248	0.1411
Word2Vec	raw	all	0.8	311	0.3248	0.3248	0.1390
Word2Vec	raw	all	0.9	309	0.3204	0.3183	0.1380
Word2Vec	raw	all	1	309	0.3204	0.3183	0.1380
Word2Vec	stem	all	0	344	0.3459	0.3826	0.1576
Word2Vec	stem	all	0.1	344	0.3459	0.3826	0.1544
Word2Vec	stem	all	0.2	344	0.3372	0.3730	0.1512
Word2Vec	stem	all	0.3	344	0.3372	0.3730	0.1510
Word2Vec	stem	all	0.4	333	0.3514	0.3762	0.1537
Word2Vec	stem	all	0.5	325	0.3600	0.3762	0.1518
Word2Vec	stem	all	0.6	323	0.3622	0.3762	0.1511
Word2Vec	stem	all	0.7	322	0.3571	0.3698	0.1502
Word2Vec	stem	all	0.8	320	0.3594	0.3698	0.1502
Word2Vec	stem	all	0.9	320	0.3594	0.3698	0.1502
Word2Vec	stem	all	1	320	0.3594	0.3698	0.1502
Word2Vec	lemma	all	0	350	0.3200	0.3601	0.1558
Word2Vec	lemma	all	0.1	350	0.3229	0.3633	0.1573
Word2Vec	lemma	all	0.2	350	0.3200	0.3601	0.1604
Word2Vec	lemma	all	0.3	349	0.3381	0.3794	0.1633
Word2Vec	lemma	all	0.4	340	0.3588	0.3923	0.1699
Word2Vec	lemma	all	0.5	335	0.3582	0.3859	0.1660
Word2Vec	lemma	all	0.6	329	0.3587	0.3794	0.1564
Word2Vec	lemma	all	0.7	324	0.3549	0.3698	0.1525
Word2Vec	lemma	all	0.8	316	0.3639	0.3698	0.1525
Word2Vec	lemma	all	0.9	316	0.3639	0.3698	0.1525
Word2Vec	lemma	all	1	316	0.3639	0.3698	0.1525

References

[1] European Environment Agency, *Circular economy in Europe developing the knowledge base*. Luxembourg: Publications Office of the European Union, 2016.

[2] M. R. Chertow, "INDUSTRIAL SYMBIOSIS: Literature and Taxonomy," *Annu. Rev. Energy Environ.*, vol. 25, no. 1, pp. 313–337, Nov. 2000.

[3] V. Hug, L. L. E. Sørensen, B. Bahn-Walkowiak, and R. Williams, "Economic Analysis of Resource Efficiency Policies," p. 97.

[4] G. van Capelleveen, C. Amrit, D. M. Yazan, and H. Zijm, "The influence of knowledge in the design of a recommender system to facilitate industrial symbiosis markets," *Environ. Model. Softw.*, vol. 110, pp. 139–152, Dec. 2018.

[5] G. van Capelleveen, C. Amrit, and D. M. Yazan, "A Literature Survey of Information Systems Facilitating the Identification of Industrial Symbiosis," in *From Science to Society*, B. Otjacques, P. Hitzelberger, S. Naumann, and V. Wohlgemuth, Eds. Cham: Springer International Publishing, 2018, pp. 155–169.

[6] R. J. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.

[7] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl.-Based Syst.*, vol. 46, pp. 109–132, Jul. 2013.

[8] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, "Collaborative Filtering Recommender Systems," *Found Trends Hum-Comput Interact*, vol. 4, no. 2, pp. 81–173, Feb. 2011.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, New York, NY, USA, 1994, pp. 175–186.

[10] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan. 2003.

[11] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[12] P. Lops, M. de Gemmis, and G. Semeraro, "Content-based Recommender Systems: State of the Art and Trends," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston, MA: Springer US, 2011, pp. 73–105.

[13] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *User Model. User-Adapt. Interact.*, vol. 12, no. 4, pp. 331–370, Nov. 2002.

[14] C. C. Aggarwal, *Recommender systems: the textbook*, 1st edition. New York, NY: Springer Science+Business Media, 2016.

[15] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, "Semantics-Aware Content-Based Recommender Systems," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 119–159.

[16] B. Kitchenham, "Procedures for performing systematic reviews," *Keele UK Keele Univ.*, vol. 33, no. 2004, pp. 1–26, 2004.

[17] C. Wohlin, "Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, New York, NY, USA, 2014, pp. 38:1–38:10.

[18] E. Agirre, D. Cer, M. Diab, and A. Gonzalez-Agirre, "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)*, Montrèal, Canada, 2012, pp. 385–393.

[19] E. Ukkonen, "On-line construction of suffix trees," *Algorithmica*, vol. 14, no. 3, pp. 249–260, Sep. 1995.

[20] M. T. Elhadi, “Text similarity calculations using text and syntactical structures,” in *2012 7th International Conference on Computing and Convergence Technology (ICCCT)*, 2012, pp. 715–719.
[21] S. Sultana and I. Biskri, “Identifying Similar Sentences by Using N-Grams of Characters,” in *Recent Trends and Future Technology in Applied Intelligence*, Cham, 2018, pp. 833–843.
[22] P. Jaccard, “Étude comparative de la distribution florale dans une portion des Alpes et du Jura,” *Bull. Société Vaudoise Sci. Nat.*, vol. 37, no. 142, pp. 547–579, 1901.
[23] S. A. Takale and S. S. Nandgaonkar, “Measuring semantic similarity between words using web documents,” *Int. J. Adv. Comput. Sci. Appl. IJACSA*, vol. 1, no. 4, 2010.
[24] L. R. Dice, “Measures of the Amount of Ecologic Association Between Species,” *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.
[25] G. Salton, A. Wong, and C. S. Yang, “A Vector Space Model for Automatic Indexing,” *Commun ACM*, vol. 18, no. 11, pp. 613–620, Nov. 1975.
[26] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513–523, Jan. 1988.
[27] G. A. Miller, “WordNet: A Lexical Database for English,” *Commun ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995.
[28] R. Rada, H. Mili, E. Bicknell, and M. Blettner, “Development and application of a metric on semantic nets,” *IEEE Trans. Syst. Man Cybern.*, vol. 19, no. 1, pp. 17–30, Jan. 1989.
[29] C. Leacock and M. Chodorow, “Combining local context and WordNet similarity for word sense identification,” *WordNet Electron. Lex. Database*, vol. 49, no. 2, pp. 265–283, 1998.
[30] Z. Wu and M. Palmer, “Verbs semantics and lexical selection,” in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, 1994, pp. 133–138.
[31] P. 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.
[32] D. Lin, “An information-theoretic definition of similarity.,” in *ICML*, 1998, vol. 98, pp. 296–304.
[33] J. J. Jiang and D. W. Conrath, “Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy,” in *ROCLING*, 1997.
[34] H. Liu and P. Wang, “Assessing Sentence Similarity Using WordNet based Word Similarity.,” *JSW*, vol. 8, no. 6, pp. 1451–1458, 2013.
[35] D. Croft, S. Coupland, J. Shell, and S. Brown, “A fast and efficient semantic short text similarity metric,” in *2013 13th UK Workshop on Computational Intelligence (UKCI)*, 2013, pp. 221–227.
[36] Y. Li, H. Li, Q. Cai, and D. Han, “A novel semantic similarity measure within sentences,” in *Proceedings of 2012 2nd International Conference on Computer Science and Network Technology*, 2012, pp. 1176–1179.
[37] J. J. Castillo and M. E. Cardenas, “Using Sentence Semantic Similarity Based on WordNet in Recognizing Textual Entailment,” in *Advances in Artificial Intelligence – IBERAMIA 2010*, Berlin, Heidelberg, 2010, pp. 366–375.
[38] G. Pirró and N. Seco, “Design, Implementation and Evaluation of a New Semantic Similarity Metric Combining Features and Intrinsic Information Content,” in *On the Move to Meaningful Internet Systems: OTM 2008*, Berlin, Heidelberg, 2008, pp. 1271–1288.
[39] H. W. Kuhn, “The Hungarian method for the assignment problem,” *Nav. Res. Logist. Q.*, vol. 2, no. 1-2, pp. 83–97, 1955.
[40] J. Z. Wang and W. Taylor, “Concept Forest: A New Ontology-assisted Text Document Similarity Measurement Method,” in *IEEE/WIC/ACM International Conference on Web Intelligence (WI’07)*, 2007, pp. 395–401.
[41] J. O’Shea, Z. Bandar, K. Crockett, and D. McLean, “A Comparative Study of Two Short Text Semantic Similarity Measures,” in *Agent and Multi-Agent Systems: Technologies and Applications*, Berlin, Heidelberg, 2008, pp. 172–181.
[42] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *J. Am. Soc. Inf. Sci.*, vol. 41, no. 6, pp. 391–407, 1990.

[43] V. Rus, N. Niraula, and R. Banjade, “Similarity Measures Based on Latent Dirichlet Allocation,” in *Computational Linguistics and Intelligent Text Processing*, 2013, pp. 459–470.
[44] D. M. Blei, “Latent Dirichlet Allocation,” *J. Mach. Learn. Res. 3*, pp. 993–1022, 2003.
[45] E. Gabrilovich and S. Markovitch, “Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis,” in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, San Francisco, CA, USA, 2007, pp. 1606–1611.
[46] P. Shrestha, “Corpus-Based methods for Short Text Similarity,” in *Rencontre des Étudiants Chercheurs en Informatique pour le Traitement automatique des Langues*, Montpellier, France, 2011, vol. 2, p. 297.
[47] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, “From Word Embeddings to Document Distances,” in *Proceedings of the 32nd International Conference on Machine Learning - Volume 37*, Lille, France, 2015, pp. 957–966.
[48] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119.
[49] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford, “Okapi at TREC-3,” in *Proceedings of TREC-3 ’95*, 1995, pp. 109–126.
[50] M. Chen, Z. E. Xu, K. Q. Weinberger, and F. Sha, “Marginalized Denoising Autoencoders for Domain Adaptation,” *CoRR*, vol. abs/1206.4683, 2012.
[51] A. Perina, N. Jojic, M. Bicego, and A. Truski, “Documents as multiple overlapping windows into grids of counts,” in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 10–18.
[52] Y. Li, D. McLean, Z. A. Bandar, J. D. O’Shea, and K. Crockett, “Sentence similarity based on semantic nets and corpus statistics,” *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 8, pp. 1138–1150, Aug. 2006.
[53] A. Pawar and V. Mago, “Calculating the similarity between words and sentences using a lexical database and corpus statistics,” *CoRR*, vol. abs/1802.05667, 2018.
[54] A. Islam and D. Inkpen, “Semantic Text Similarity Using Corpus-based Word Similarity and String Similarity,” *ACM Trans Knowl Discov Data*, vol. 2, no. 2, pp. 10:1–10:25, Jul. 2008.
[55] M. Islam and D. Inkpen, “Second Order Co-occurrence PMI for Determining the Semantic Similarity of Words,” in *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC’06)*, Genoa, Italy, 2006.
[56] R. Mihalcea, C. Corley, and C. Strapparava, “Corpus-based and Knowledge-based Measures of Text Semantic Similarity,” in *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, Boston, Massachusetts, 2006, pp. 775–780.
[57] H. H. Vu, J. Villaneau, F. Saïd, and P.-F. Marteau, “Sentence Similarity by Combining Explicit Semantic Analysis and Overlapping N-Grams,” in *Text, Speech and Dialogue*, Cham, 2014, pp. 201–208.
[58] C.-Y. Lin and E. Hovy, “Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics,” in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, Stroudsburg, PA, USA, 2003, pp. 71–78.
[59] D. Bär, C. Biemann, I. Gurevych, and T. Zesch, “UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures,” in *Proceedings of 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*, Stroudsburg, PA, USA, 2012, pp. 435–440.
[60] F. Šarić, G. Glavaš, M. Karan, J. Šnajder, and B. D. Bašić, “TakeLab: Systems for Measuring Semantic Text Similarity,” in *Proceedings of 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, Stroudsburg, PA, USA, 2012, pp. 441–448.
[61] C. Banea, S. Hassan, M. Mohler, and R. Mihalcea, "UNT: A Supervised Synergistic Approach to Semantic Text Similarity," in *Proceedings of 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*, Stroudsburg, PA, USA, 2012, pp. 635–642.
[62] G. Soğancıoğlu, H. Öztürk, and A. Özgür, "BIOSSES: a semantic sentence similarity estimation system for the biomedical domain," *Bioinformatics*, vol. 33, no. 14, pp. i49–i58, Jul. 2017.
[63] C. Wang, L. Long, and L. Li, "HowNet based evaluation for Chinese text summarization," in *2008 International Conference on Natural Language Processing and Knowledge Engineering*, 2008, pp. 1–6.
[64] C. Zhao, X. Yao, and S. Sun, "A HowNet-based Feature Selection Method for Chinese Text Representation," in *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, 2009, vol. 1, pp. 26–30.
[65] O. Bodenreider, "The Unified Medical Language System (UMLS): integrating biomedical terminology," *Nucleic Acids Res.*, vol. 32, no. 90001, pp. 267D – 270, Jan. 2004.
[66] J. D. Ferreira and F. M. Couto, "Semantic similarity for automatic classification of chemical compounds," *PLoS Comput. Biol.*, vol. 6, no. 9, 2010.
[67] F. Benedetti, D. Beneventano, S. Bergamaschi, and G. Simonini, "Computing inter-document similarity with Context Semantic Analysis," *Inf. Syst.*, vol. 80, pp. 136–147, Feb. 2019.
[68] L. Burnard, "Reference Guide for the British National Corpus (XML Edition)," 2007.
[69] W. N. Francis and H. Kucera, *The Brown Corpus: A Standard Corpus of Present-Day Edited American English*. 1979.
[70] Q. Chen, S. Kim, W. J. Wilbur, and Z. Lu, "Sentence Similarity Measures Revisited: Ranking Sentences in PubMed Documents," in *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, New York, NY, USA, 2018, pp. 531–532.
[71] K. Sugathadasa *et al.*, "Synergistic union of Word2Vec and lexicon for domain specific semantic similarity," in *2017 IEEE International Conference on Industrial and Information Systems (ICIIS)*, 2017, pp. 1–6.
[72] Sepa, "Guidance on using the European Waste Catalogue (EWC) to code waste," no. November, 2015.
[73] P. Black, "Ratcliff/Obershelp pattern recognition," *Dictionary of Algorithms and Data Structures [online]*, Dec-2004. [Online]. Available: https://xlinux.nist.gov/dads/HTML/ratcliffObershelp.html. [Accessed: 05-Mar-2019].
[74] M. Deshpande and G. Karypis, "Item-based top-N Recommendation Algorithms," *ACM Trans Inf Syst*, vol. 22, no. 1, pp. 143–177, Jan. 2004.