Universiteit
Leiden

Master Computer Science

Process Extraction from Scientific Literature

Name:	Xiaoling Zhang
Student ID:	s1966367
Date:	20th August,2019
Specialisation:	Computer Science and Advanced Data Analytics
1st supervisor:	Wessel Kraaij
2nd supervisor:	Bernhard Steubing

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Process Extraction from Scientific Literature

An Unsupervised Approach Based on Distributional Semantics and Co-occurrence for Process Extraction on Biorefining Scientific Literature[*]

Xiaoling Zhang
Faculty of Science, Leiden University
Leiden, Netherlands
x.zhang.25@umail.leidenuniv.nl

ABSTRACT

Technologies are developing at a rapid pace, resulting in an ever-increasing amount of digital text data available online. Lots of information are hidden in the digital data, which needs to be mined with data mining methods. Text mining methods are widely used for information extraction task. Process extraction, similar to one of the text mining tasks, relation extraction, has gained more attention in the last decade. The elements of the process that need to be extracted can be slightly different for different domains. However, the main components of a process model can be loosely defined as a structured list of descriptors, consisting of inputs, a transformation step and outputs. Some researches have been brought to extract processes from material synthesis procedure texts [18]. Apart from material synthesis procedures, processes in other domains are also described in scientific literature, such as biorefineries. Biorefining, a crucial technique in the circular economy, has been gaining more importance in recent years and the biorefinery processes in texts can be useful for finding new pathways, building process models, supporting decision making when several pathways are available, etc. In the biorefinery context, the process model is instantiated with feedstock, technology and products. In this work, we show how simplified biorefinery process models, consisting of inputs, outputs and technologies only, can be extracted from texts and we have constructed a text mining pipeline to extract biorefinery processes. Furthermore, a baseline method and a distributional semantics-based method, utilizing different word embedding models have been implemented, following the pipeline to extract biorefinery processes from abstracts for articles. The performances of these two methods have been evaluated and compared. An expert-annotated seed list was used for the baseline method, while the list is expanded using the word embedding models in the distributional semantics-based method. It was found that the distributional semantics-based method is capable of extracting processes, given the definition that processes are in the form of triplets (input, technology, output) in sentences. Meanwhile, the results of our experiments show that the distributional semantics-based method can achieve better performance than the baseline method for the process extraction task.

CCS CONCEPTS

• **Text mining → Domain-specific process extraction**; *Methods*; • **Domain-specific process extraction** → Domain-specific characters; • **Methods** → Co-occurrence and distributional semantics;

KEYWORDS

biorefinery, process extraction, co-occurrence, word embedding, entity extraction

1 INTRODUCTION

1.1 Background

1.1.1 Text Mining. With the rapid development of digital technology in the last few decades, a surge of data in forms of structured, unstructured and semi-structured sets are generated at a high speed and with large quantities of valuable information hidden inside. Many decision-making methods based on intuition and traditional experience have been replaced by data analysis, data mining, etc. as a result of the rapid development of big data technologies [12]. Data mining is a mature technology which has been widely studied to discover unknown patterns from structured numerical information [35].

However, by some estimates, more than 80 percent of contemporary data is stored as unstructured or semi-structured data [30]. Text, often described as unstructured data, is generated every day in the form of news articles, research publications, blogs, question answering forums, and social media [20]. Due to the huge amount of textual data, it is necessary to develop techniques which extract information automatically from documents. The implicit and explicit information that is extracted from texts can then help access and manage hidden knowledge in the large text corpora [20].

Text mining, also known as Knowledge Discovery from Text (KDT), is the process of discovering knowledge from large text corpora [31]. It is a research area which applies techniques such as data mining, natural language processing (NLP) and knowledge management [28]. During the past years, this has grown significantly as a research area, and techniques have been developed for information extraction (IE) from text data and text summarization, etc.; unsupervised and supervised learning methods have been proposed for mining information from texts such as Latent semantic indexing (LSI) and transfer learning with text data [1]. Text mining techniques are continuously applied in search engines, email filters, fraud detection, etc. in industry, while people in academia apply it to publications or scientific literature from biomedical, legal and chemical domains to extract valuable information [7] [8] [28] [33].

Process extraction, quite similar to one of the main text mining tasks (relation extraction), usually performed after the named entity recognition task, is gaining more importance today in business and material synthesis domains [1] [18]. Various approaches such as supervised methods, rule-based methods, pattern-based methods,

[*] Text mining applied for extracting processes hidden in textual descriptions

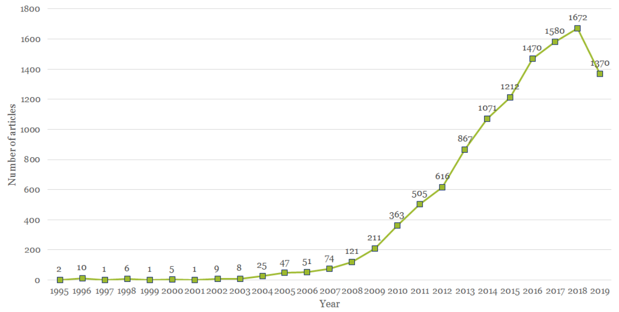

Figure 1: Number of results for the Boolean query $'biorefinery'\ AND\ ('technology'\ OR\ 'process')$ through Elsevier's API

co-occurrence methods, etc. have been studied. To extract relations and processes from texts, entity extraction is a crucial step. Existing approaches for extracting entities from texts require either a lot of training data or rules generated by experts. Word embedding models, such as GloVe, word2vec and fastText [4] [17] [21], are becoming more and more popular today for recognizing similar terms that occur in the same context without a huge amount of human efforts.

1.1.2 Biorefinery Domain. Bioeconomy is said to be becoming increasingly important globally for economic and environmental sustainability [3]. It is believed that in the coming decades, a transition to a carbon-neutral, sustainable, circular bioeconomy (CE) must be achieved to avoid potentially catastrophic climate change and other consequences[24]. Thus, research on the bioeconomy, an economy where the basic building blocks for materials, chemicals, and energy are derived from renewable biological resources, has been steadily increasing in recent decades [16]. Many currently emerging technologies, such as bio-, energy-, and information technologies, will be decisive in achieving climate and other environmental targets. Biorefinery, the sustainable process of converting biomass into a spectrum of marketable products such as energy and other beneficial products like chemicals, is a crucial technology strategy for bioeconomy [25]. Due to this, a growing number of studies have been focused on biorefinery and a comprehensive body of literature (thousands of articles, patents, and other online resources) has been created in the last decades. Fig.1 shows the number of articles collected in each year since 1995 for the Boolean query $'biorefinery'\ AND\ ('technology'\ OR\ 'process')$ through Elsevier's API. This data provides a brilliant basis for distilling information such as the optimal use of biomass resources, technologies, and conversion pathways for replacing petrochemical products in a bioeconomy.

Life Cycle Assessment (LCA), increasingly used for emerging technologies, is the method for assessing the environmental performance of products or services across the refining life cycle and supporting decision making for technology design, industry, research, etc [19]. The LCA is based on detailed descriptions of the inputs (e.g. energy and materials) and outputs (e.g. products, wastes, and emissions) of individual processes (e.g. steel and electricity production) in the so-called life cycle inventories. The individual process (Life cycle) inventories are stored in an LCA database, however, a lot of LCA data published in scientific articles does not find its way into LCA databases. Fig.2 shows the typical workflow of a unit process which mainly contains feedstocks (inputs), technologies (operations) and products (outputs). These data form the fundamental elements in an LCA database.

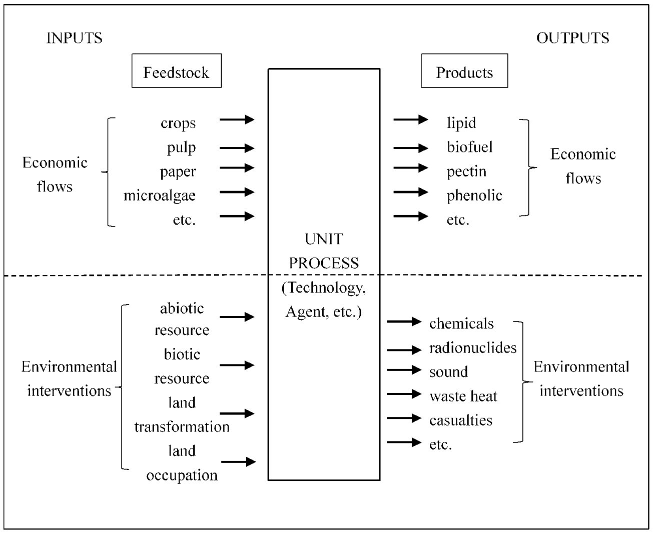

Figure 2: Basic structure of a unit process

Though it is feasible to manually extract LCA data from limited scientific articles, a lot more articles need to be read to obtain valuable information. However, it becomes a time-consuming and laborious task to extract structured LCA data from unstructured texts when the number of articles grows large. Therefore, text mining methods are required to automatically extract LCA data from textual documents . The LCA data can subsequently be used for biorefinery process model integration and new pathway exploration, giving new options to people of both industrial and academic backgrounds. However, extracting all LCA data is a complicated task, thus it is meaningful to first extract simplified process. This would enable us to see which kinds of technologies and products are covered in the literature and possibly how these could be connected to supply chains.

1.2 Related Work

1.2.1 Text Mining Methods. Process extraction relates closely to relation extraction problems, which have been investigated for different domains. These problems are generally addressed by constructing a natural language processing (NLP) pipeline, which includes sentence splitting, tokenization, POS-tagging, sentence parser and named entity recognition [1] [2] [10] [36]. In recent years, many approaches based on the described NLP pipeline have been proposed to extract relations from texts in different domains, ranging from co-occurrence approaches to machine learning-based approaches. These approaches are different in many aspects, such as the way in which NLP techniques are used for analyzing the input text and which methods are used to learn extraction rules. For example, it could be either a manually defined pattern or automatically learned from training datasets [5] [15].

The simplest co-occurrence approach for relation extraction tends to achieve high recall but low precision. This approach is used for protein-protein interactions extraction on AIMed corpus where the result shows that only 17% of protein pairs that belong to the same sentence actually describe protein-protein interactions[22]. Pattern-based approaches, relying on automatically generated (bootstrapping or directly from corpora) or manually defined patterns (a time-consuming process where domain expert are involved), are used to extract relations between two proteins based on word forms which usually use regular expressions to express patterns or syntactic analysis of a sentence [34]. The rule-based approaches are quite similar to the pattern-based approaches since the rules are either manually defined or automatically generated from training data. However, rule-based approaches are more flexible than pattern-based approaches since they are built at a more abstract level. For example, syntactic structures, using semantic relations of the sentence and grammatical relations, are used in rule-based approaches to extract relations using a set of heuristic rules rather than given set patterns. [9][23]. The rule-based method has been used successfully with several types of texts. On expert-annotated materials science journal articles, it was used for the automatic extraction of structured representations of synthesis procedures [18]. On biomedical literature, protein-protein interactions have been extracted [36]. Furthermore, business process models were extracted from existing documentation within the organization through the grammatical structure of a sentence [1]. Except for the aforementioned approaches, machine learning-based (ML-based) approaches, including feature-based and kernel-based approaches, have developed significantly over the last decades for domains in which annotated data is available, such as in the biomedical domain. Supervised learning is most used in ML-based approaches, in which relation extraction tasks are regarded as classification problems [13] [29].

The approaches described can be used for relation extraction or similar tasks in both domain-specific and general tasks. However, domain-specific tasks may need further specific assumptions since each domain has its own characteristics.

In terms of word embedding models, multiple techniques are proposed to use unlabeled data to learn word representation where words in a sentence are mapped into low-dimension space vectors. The state-of-the-art techniques used for generating word embedding models are GloVe, word2vec, and fastText. GloVe is capable of capturing statistical information from the corpus as it is trained on nonzero elements in the word-word co-occurrence matrix [21]. Word2vec is developed for computing continuous vector representations of words from large data sets with two alternative model architectures, namely CBOW (Continuous bag of words, which predicts the target word based on its context) and SG (Skip-gram in which the target word is given for predicting the near words). Word2vec word embedding models, used for automatically extending Knowledge Base facts, are trained using one of these two models [17]. FastText is an extension of word2vec, which is said to be capable of capturing subword information and thus improves the quality of vector representations for words [4]. GloVe, word2vec, and canonical correlation analysis (CCA) word embedding methods were studied by Miran et al. for the named entity recognition task and it was found that nearest neighbors can be used to describe the word expression ability of word embedding [27]. For example, word embedding models are used for recognizing drug names from biomedical texts [26].

1.2.2 Applications in the Biorefinery Domain.

Steadily increasing researches on the circular economy, especially for the central technique (biorefinery), leads to more attention to the problem of how an automated data analysis approach can be built and can help in analyzing large volumes of text. Using a robust manually created list of TAPs (Technologies, Applications and Products) and a list of wastes in combination with topic modeling and co-occurrence analysis of TAPs and wastes from articles, Chris B. Davis et al. presented a method of distilling and presenting potential value pathways for implementing the Circular Economy [6]. A semantic approach was proposed for integrating the biorefinery process model and data through ontology engineering where the process models and data are semantically annotated as input(s), output(s), precondition(s), etc. [14]. Nikolaos et al. also introduced an ontology-controlled input-output matching model integration framework based on an existing framework [32].

To the best of our understanding and according to the related work that we have reviewed, there is no research focusing on automatically extracting the biorefinery process yet. Therefore, inspired by the related work and the state-of-the-art technologies, we provide a definition of how a biorefinery process can be extracted from texts and propose an unsupervised distributional semantics-based approach for process extraction from biorefinery texts with limited training data. In this thesis, we do not aim at extracting the full information that makes up unit processes. Instead, the intention is to extract the key inputs and outputs of a biorefining technology, as well as the name of this technology.

1.3 Challenges and Main Contributions

We can conclude from Section 1.2 that though significant strides have been made in relation extraction tasks in various domains and computer-assisted data analysis approaches have been applied in the domain of biorefining, there is no method for extracting biorefinery processes from texts. In terms of the approaches mentioned in Section 1.2, it is found that co-occurrence methods tend to give high recall but suffer low precision; rule-based methods and pattern-based methods tend to return results in limited patterns (low recall) while supervised methods achieve better performance but require lots of training data[5]. To extract the biorefinery process from texts, we are confronted with the following challenges:

- There is no standard procedure yet for extracting biorefinery processes and it is unclear how extracted biorefinery processes should look like.
- Only a limited number of domain-specific entities are available, which means we have only little training data and it is not feasible to use supervised methods.

To extract the biorefinery process, with the aforementioned challenges in mind, we provide a definition of how a biorefinery process can be extracted from texts in Section 2 and propose an unsupervised distributional semantics-based approach for process extraction from biorefinery texts with little training data in Section 3. The main contributions of this thesis are:

- A definition of a formal abstract metamodel for a biorefinery process.
- A baseline method based on dictionary-based look-up entity extraction for process extraction from texts using a ground truth seed list.
- A novel approach for biorefinery process extraction utilizing distributional semantics with little training data.
- Evaluation on and comparison between both the baseline method and the distributional semantics-based approach for process extraction.

2 PROBLEM STATEMENT

We aim to extract structured representations of biorefinery procedural text, as described in the scientific literature, describing the biorefinery process. As mentioned in Section 1, there are no studies focused on this topic. Thus, a definition of the biorefinery process metamodel, with the problem of how biorefinery processes can be extracted from scientific literature in mind, is needed. With the defined metamodel, the subsequent task of this work is to build and evaluate a text mining pipeline for extracting the process from abstracts with little training data. Since only little training data is available, a method for expanding the existed training data and utilizing it to extract process from texts is needed. Word embedding models are a candidate method to overcome the labeled data sparsity problem. However, the precise configuration of word embedding models for this task in an open challenge. The research questions and the hypotheses for this work are given in Section 2.1, the definition of the biorefinery process metamodel for extracting process is then described in Section 2.2. Lastly, the way of vector representation of terms describing the biorefinery process using word embedding models is explained in Section 2.3.

2.1 Research Questions and Hypotheses

To address the above-mentioned challenges, the following research questions are studied:

- Research Question 1:
  How can text mining be used to extract process models in the context of biorefining technology?
- Research Question 2:
  How cen text mining pipeline be built for extraction of the biorefinery processes and evaluated with little training data?

Regarding the research questions, we made the following modeling assumptions in order to create a meaningful but simplified text mining task. Several hypotheses were defined to guide our research.

- Assumption 1:
  A biorefinery process is described in a sentence.
- Assumption 2:
  Word embedding models are capable of expanding the size of the training data for extracting the biorefinery process.
- Hypothesis 1:
  Self-trained word embedding models perform better for expanding the training data.
- Hypothesis 2:
  Self-trained word embedding models achieve better performance for process extraction tasks than the pre-trained word embedding models with the self-trained models being more capable of capturing domain-specific semantics features.
- Hypothesis 3:
  Distributional semantics-based methods with word embedding models obtain better recall, but lower precision than a dictionary lookup-based baseline method for the process extraction task. This is because the training data for process extraction is expanded by using word embedding models, however, not all triplets extracted by using the expanded training data are valid.

2.2 Definition of A Process Metamodel

In the upper half of Fig.2, an example of the typical workflow of a unit process in terms of economic flow in the biorefinery life cycle is depicted. The life cycle contains mainly feedstocks (inputs), technologies (operations) and products (outputs). Therefore, the task of extracting a biorefinery process can be defined as extracting the input, output and the technology used in the corresponding process. To extract a process from texts, we assume that a triplet exists for a process. The triplet is defined as (I, T, O) in a sentence (I stands for input, O represents the output and T means the technology used in a process).

2.3 Vector Representation of Terms

Word embedding, useful numerical representations of words with dense vectors, is capable of capturing syntactic and semantic similarities with other words and contexts of words in a document. With the intuitive assumption that words have a similar meaning if they are in the same context, word embedding is used to find similar terms to expand the biorefinery process description terminology list. Word embeddings can be used as word representation, phrase representation, and document representation. For words that have a vector representation in the word embedding model, the word can be represented by E(x). E(x) is the word embedding for word x, such as E('biorefinery'), which represents the word embedding for the term 'biorefinery'. In terms of embedding for phrases, a common method is to add up the word embedding for the words in the phrase. This method is used in this work to calculate the word embedding for phrases. Take the phrase 'methanol production' as an example, E('methanol production') = E('methanol') + E('production').

3 METHOD

With the definition of the biorefinery process metamodel given and the vector representation of terms defined in Section 2.2, we built a text mining pipeline for extracting the biorefinery process information from abstracts. The workflow of our approach is illustrated in Fig3. This pipeline includes both the baseline method and the distributional semantics-based approach. The distributional semantics-based approach is an extension of the baseline approach,

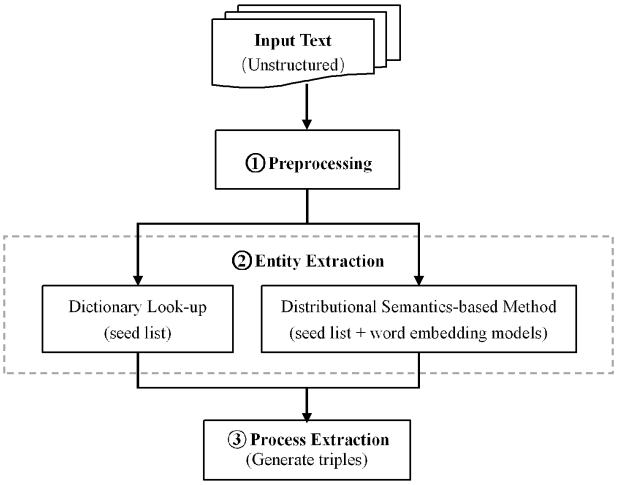

Figure 3: Workflow of the text mining pipeline for process extraction

since the difference between these two approaches is that the distributional semantics-based approach used the word embedding model to expand the seed list which is then used for process extraction. The reason why word embedding models are considered for extracting entities and process extraction in this work is that word embedding models are becoming more and more popular for recognizing similar terms that occur in the same context without large amount of human effort. The pipeline starts with data collection followed by data preprocessing, mainly including document retrieving, text segmentation, punctuation removal, stopwords removal and noun-phrase chunking. The next step is to extract the entities from texts with entity extraction methods. In this step, merely the seed list is used for entity extraction in the baseline method while the expanded seed lists, obtained by using word embedding models, are used for entity extraction in the distributional semantics-based method. With the detected entities, process extraction is performed to extract instances which describe biorefinery processes. Methods studied in this work are described in detail in the following subsections.

3.1 Data Collection

3.1.1 Articles Related to Biorefinery Technology or Biorefinery Process. The articles were collected via API provided by ScienceDirect[1] through Elsevier, with full-text retrieval query $biorefinery\ AND\ (\ technology\ OR\ process)$ which aimed at locating articles containing biorefinery processes from the years 1995 to 2019. With this query, we collected 11,298 distinct matching results in XML (Xtensible Markup Language) format from which contents could be easily extracted. However, only 8,477 articles are full-text articles and 9,831 articles contain titles, keywords, and abstracts among the collected dataset due to the limitations on automated full-text downloads from ScienceDirect.

3.1.2 Gold Standard Entity List / Seed List. To generate the seed list for process extraction, 19 articles with titles and abstracts from the whole retrieved dataset were annotated by a domain expert with the help of the web-based text annotation tool BRAT[2]. The entities were labeled as feedstock (input,i), technology (t) or product (output, o) through annotation since the terms used for describing a biorefinery process are mainly included in these three categories. A list of 150 entities with its label, consisting of 39 feedstock terms, 54 technology terms and 57 terms which stand for products, is generated for later tasks.

3.2 Preprocessing

Preprocessing is a fundamental step in all kinds of NLP tasks, transferring texts from human language to a machine-readable format for further processing, since the datasets are often quite messy with a lot of noisy data. The operations in the preprocessing step could vary in different NLP tasks and the data used in this work are preprocessed slightly differently, with regard to the specific tasks.

3.2.1 Text Segmentation. For all tasks in this work, text segmentation is performed to obtain meaningful language units, such as sentences and words. Firstly, XML parsing with ElementTree[3], markup noise, non-standard characters removal with regular expressions are performed to get cleaned texts from XML files. Considering the entity extraction task, where a period could appear in both entities and the end of a sentence, sentence splitting is then used to split the document into sentences before tokenization. After sentence splitting and tokenization, the lower casing step is performed on tokens and numerical values are removed. The numerical values are removed here since only little terminologies for inputs, outputs are chemical compounds and they are ignored in this work.

3.2.2 Punctuation and Stopwords Removal (Optional for some tasks). Except for noun-phrase chunking based entity extraction, punctuation and stopwords are removed for all tasks in this thesis, as punctuation and stopwords make sense in shallow parsing while it is meaningless for other tasks.

3.2.3 Part-of-Speech(POS) tagging. POS tagging can be useful for recognizing unigram entities by their linguistic characteristics, for example, most of the terms for describing biorefinery processes are nouns (for technology, feedstock and product) and verbs (terms for technology description). Thus, part-of-speech generated from POS tagging with the NLTK toolkit is used for filtering out words in these classes.

3.2.4 Noun Phrase Chunking. Terminologies for biorefinery process description, such as screw pressing, selective enzymatic liquefaction, yeast oil, often contains more than one word. Therefore, the noun phrase chunking step is performed in order to recognize and extract those terms. The NLTK toolkit[4], can recognize noun phrase chunks based on POS tags, is used to collect noun phrases with predefined chunk grammars in which the rules for how sentences should be chunked are given. To get the chunking sequences from texts, the following chunk grammar is applied to

[1] https://dev.elsevier.com/documentation/FullTextRetrievalAPI.wadl

[2] https://brat.nlplab.org/

[3] https://pypi.org/project/elementtree/

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

get chunking sequences: $< JJ.* > * < NN.* >$ (JJ means adjective and NN means noun). With this grammar, sequences which follow the rule that any number of adjectives of any types, followed by one or more nouns of any types, are grouped into noun phrases. All noun phrases detected are limited to three words since most terminologies for biorefinery process description do not exceed three words.

3.3 Entity Extraction

Entity extraction is the process of identifying specific types of entities within texts. In this section, two entity extraction methods will be introduced. The first one is the dictionary lookup-based baseline method, which extract entity by match the entity in the sentence to the seed list. The other one is the distributional semantics-based method in which entities are extracted by computing the similarity score between word embedding representation for newly detected entities and entities in the seed list.

3.3.1 Dictionary-based Look-up method (Baseline method). In this method, the seed list is used for extracting entities. The noun phrases in sentences are extracted according to description in section 3.2.4 and then the dictionary-based look-up is performed in order to identify the entity which matches that in the seed list.

3.3.2 Bootstrapping Entity Extraction. As only a limited number of entities are available in the seed list and it is obvious that only entities from the list can be extracted using the baseline method, a method for expanding the seed list is needed. To expand the seed list, POS tags and noun phrase chunking are performed on all abstracts from retrieved articles (except for the 19 abstracts which is used for generating seed list) to generate all possible entity candidates ($i_1, i_2, ..., i_n$), including nouns, verbs, and noun phrases since some of the technologies may be described in the verb form. Then, as described in Section 2.3, word embedding for words and phrases are computed for each entity in the seed list and entity candidates extracted in the former step respectively. Since the entities in the seed list are categorized into three categories, the average embedding for each category is calculated using the following equation 11. The k in the equation 11 represents the number of entities in that specific category.

$$E(X) = \frac{\sum_{i=1}^{k} E(X_i)}{k} \tag{1}$$

E(I), E(T) and E(O) are the average word embedding for the entities in feedstock/input (I) list, the technology(T) list and the products/output (O) list respectively. As described in Algorithm 1, for all entity candidates in ($i_1, i_2, ..., i_n$), the similarity between an instance and every average embedding is computed. The instance is assigned to the category X ($X \in (I, T, O)$) whose similarity (See equation 2) is higher or equal to a threshold τ_{sim}. The $NumberOfIterations$ needs to be set before the algorithm runs, otherwise it would run until all the candidates are assigned to different categories. This is meaningless since the entities that are assigned to the seed list should be at high relevance to the biorefinery process description terminologies. (The threshold and the number of iterations is set to 0.8 and 5, respectively. These parameters are selected according to the performance of the experiments.)

```
Input: $Candidates$ = $i_1$, $i_2$, ..., $i_n$
Input: $AverageEmbeddings$ = E($I$), E($T$), E($O$)
Input: $NumberOfIterations$
Output: $CategoryList$ = $I$, $T$, $O$
while $NumberOfIterations$ do
    for $i_n \in Candidates$ do
        for $E(X) \in AverageEmbeddings$ do
            if $Sim(E(i_n), E(X)) \geq \tau_{sim}$ then
                $X = X \cup i_n$
            else
                pass
```

Algorithm 1: Bootstrapping Entity Extraction

3.4 Process Extraction

To extract the process description from texts, a co-occurrence method is performed on each sentence in the text. In this method, we assume that a biorefinery process description can be extracted, if entities that are in I, T, O categories co-occur in a sentence. An example of how a process is extracted from an abstract is shown in Fig.4. Starting with an abstract which is split into sentences, the named entities in each sentence are recognized and assigned to an entity category by applying one of the methods described in Section 3.3. The entities which are recognized as both inputs and outputs are kept in the entity list in the entity extraction step. For the dictionary lookup-based baseline method, the triplet is generated by firstly performing the entity extraction process described in Section 3.3.1, and then generate triples for for each sentence, if input, technology and output co-occur. Meanwhile, in the distributional semantics-based method, the triplets in a sentence are generated as described in Algorithm 2. The process described in a sentence can be constructed by aggregating all the entities to the corresponding category. The last box in Fig.4 shows a process extracted from a sentence.

```
Input: $I$ (feedstock terms in a sentence) = $I_1$, $I_2$, ..., $I_n$
Input: $T$ (technology terms in a sentence)= $T_1$, $T_2$, ..., $T_n$
Input: $O$ (product terms in a sentence)= $O_1$, $O_2$, ..., $O_n$
Output: $triplets$ (list of ($i$, $t$, $o$) triplets)
initiate triplets = []
for $I_n \in Flist$ do
    for $T_n \in Tlist$ do
        for $O_n \in Plist$ do
            if ($I_n$, $T_n$, $O_n$) and ($O_n$, $T_n$, $I_n$) not in triplets then
                triplets = triplets $\cup$ ($I_n$, $T_n$, $O_n$)
            else
```

Algorithm 2: Generate triplets for entities co-occur in a sentence

4 EXPERIMENT SETUP

4.1 Experiment Environments

All experiments are run on a machine which has a 2.80 GHz Intel Core i7-7700HQ processor with 16GB of RAM, a 512GB M.2 PCIe SSD, an Nvidia GeForce GTX 1050 Ti Max-Q GPU with 4GB of VRAM and an Intel UHD Graphics 630 GPU. Python (64 bit) v3.6.8 is used for all experiments and some of the main libraries involved in this work are listed as follows:

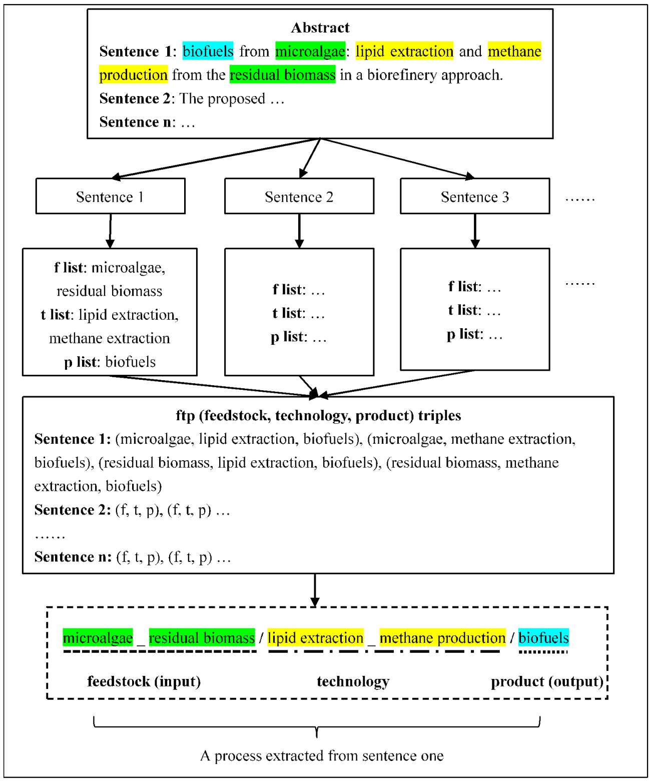
Abstract
Sentence 1: biofuels from microalgae: lipid extraction and methane production from the residual biomass in a biorefinery approach.
Sentence 2: The proposed …
Sentence n: …
Sentence 1
Sentence 2
Sentence 3
……
f list: microalgae, residual biomass
t list: lipid extraction, methane extraction
p list: biofuels
f list: …
t list: …
p list: …
f list: …
t list: …
p list: …
……
ftp (feedstock, technology, product) triples
Sentence 1: (microalgae, lipid extraction, biofuels), (microalgae, methane extraction, biofuels), (residual biomass, lipid extraction, biofuels), (residual biomass, methane extraction, biofuels)
Sentence 2: (f, t, p), (f, t, p) …
……
Sentence n: (f, t, p), (f, t, p) …
microalgae _ residual biomass / lipid extraction _ methane production / biofuels
feedstock (input)
technology
product (output)
A process extracted from sentence one

Figure 4: An example of a process extracted from text

- Elementtree v1.2.6-20050316
- Regex 2019.04.14
- NLTK (Natural Language Toolkit) v3.4.1
- Gensim v3.4.0
- Numpy v1.15.14
- Scikit-learn v0.20.1

A more detailed description of how the libraries are used in this work could be found in Appendix B.1

4.2 Word embedding models

As mentioned in previous sections, the state-of-the-art word embedding models including GloVe, word2vec, and fastText are used in the entity extraction task and the process extraction task in our approach. There are some pre-trained models available for all these models at gensim-data[5]. Though it would be highly efficient to use pre-trained models to get the word embedding for words in our corpus, not all terms could find its word embedding in the pre-trained models. Therefore, we not only experiment with the pre-trained models but also trained our own word embedding models on our own domain-specific corpus as well as using Wikipedia articles downloaded at Wikimedia dump [6] in this work. The word embedding models used in our experiments are illustrated in Table 4.

4.3 Evaluation Methods and Experiments

Quantitative evaluation metrics including *accuracy*, *precision* and *recall*, *F score*, and *similarity score* are used to evaluate the performance of our methods. For the entity extraction task, the *accuracy* and the *similarity score* are used to describe the quality of the entities extracted by using word embedding models. Meanwhile, *precision*, *recall* and *F score* are computed to evaluate the performance of the process extraction task. The definitions of these evaluation metrics are given in Appendix B.2.

Since the dictionary-based lookup method for the entity extraction task is merely detecting entities that are in the seed list, all entities in the seed list will be extracted from texts and there is no need to evaluate the baseline method. However, the entities in the seed list are used for expanding the entity list in the distributional semantics method, which is described in Algorithm 1. To evaluate the performance of the distributional semantics-based method for the entity extraction task, all models listed in Table 4 are used to generate entities. Then, we get the corresponding vector representation for the extracted entities and calculate its cosine similarity score with entities in the seed list. A higher similarity score means that the entities are more likely to occur in the same context, sharing either syntactic or semantic features. In this respect, accuracy means the number of relevant entities divided by the number of all entities extracted by using the word embedding-based method for entity extraction. However, it is unfeasible to evaluate all the extracted terms since the expanded entity list usually contain thousands of entities, which would cost too much time evalute all the extracted entities. Thus, for all extracted entities, with an expert from biorefinery domain, we randomly sampled 30 entities to evaluate the accuracy of the entity extraction task in terms of different word embedding models.

For the process extraction task, the seed list is used in the baseline method and different word embedding models from Table 4 are selected to extract processes in the distributional semantics-based method as described in Algorithm 2. For both methods, we calculated the number of abstracts which contain triplets in order to see if the distributional semantics method is capable of extracting processes with little training data. However, the number of abstracts which contain triplets are not convincing enough for proving which method is better, since some extracted triplets can be invalid. Therefore, we randomly select 50 abstracts where triplets can be extracted using our embedding models and then compute the precision and recall for the evaluation of the process extraction task. In order to compare the performance of different embedding models, the randomly selected abstracts are chosen from the abstracts where triplets can be extracted for all selected word embedding models.

5 RESULT AND DISCUSSION

In this section, results for both the baseline method and the distributional semantics-based method will be discussed. The evaluation metrics used are described in Section 4.3 and the word embedding models used for experiments are listed in Table 4. The results will be discussed in the following aspects in accordance with the experiments described in Section 4.3: pre-trained models and self-trained models for entity extraction task; pre-trained models and self-trained models for process extraction task; baseline method and

[5] https://github.com/RaRe-Technologies/gensim-data
[6] https://dumps.wikimedia.org/

our distributional semantic-based method for process extraction task.

5.1 Entity Extraction

For the entity extraction task, as described in Section 4.3, we used all word embedding models listed in Table 4 to extract entities and then compare the similarity (see equation 2) of extracted terms and the accuracy (see equation 3) of different word embedding models. The analysis of how pre-trained models and self-trained models differ from each other will be given after the results are illustrated.

5.1.1 Results. In order to get an idea of how word embedding models can be used to distinguish terms and evaluate the entity extraction task in a qualitative way, the cosine similarity scores for term pairs ('bioethanol', 'ethanol'), ('gasification', 'ethanol') and ('man', 'ethanol') are calculated for all embedding models used in our experiments. The terms for calculating the cosine similarity scores are chosen with the following reasons, 'ethanol' and 'bioethanol' are typical outputs in biorefinery processes, whereas 'gasification' represents the technology used in a biorefinery process. Except for those two terms, the purpose of choosing the term 'man' is to check if the models are able to distinguish domain-specific terms from general terms. As shown in Table 1, the similarity scores between 'bioethanol' and 'ethanol' are higher than the similarity scores between 'gasification' and 'ethanol' while 'man' obtains the lowest similarity scores for all models. Meanwhile, the similarity score for the same term pairs varies from model to model. It is found that the fastText-Wiki12-OD-300 model attains the highest similarity score for both the ('bioethanol', 'ethanol') and ('gasification', 'ethanol') term pairs while the highest similarity score for ('man', 'ethanol') is obtained using the fasttext-wiki-news-subwords-300 model. For all pre-trained models, the fasttext-wiki-news-subwords-300 model and the GoogleNews-vectors-negative300 model achieve higher similarity scores for all three term pairs while the CBOW-Wiki12-OD-win2-300 model achieves the second highest similarity scores for ('bioethanol', 'ethanol') and ('gasification', 'ethanol') term pairs.

In terms of the accuracy of extracted entities, some representative models are chosen from models listed in Table 1 and the accuracy for the selected models are given in Table 2. The pre-trained GoogleNews-vectors-negative300 model is the most accurate one among all the models and the self-trained CBOW-OD-win2-300 model achieves relatively high accuracy among the rest of the models, while the SG-OD-win2-300 model only get an accuracy of 33.33%. The self-trained fastText-Wiki12-OD-300 model and the fasttext-wiki-news-subwords-300 get an accuracy of 29.68% and 33.33%, respectively. Comparing to those models, the pre-trained GloVe models just attain ordinary results.

5.1.2 Discussion – Self-trained Models and Pre-trained Models for Expanding Training Data. It is found that the word embedding models are capable of expanding the seed list with regard to results shown in both Table 1 and Table 2, since the cosine similarity for domain-specific terms are much higher where some of the cosine similarity scores for the ('man', 'ethanol') term pair are even less than 0, indicating that the two vectors are opposed. The best accuracy achieved is 76.67% by using the GoogleNews-vectors-negative300 model. It is observed that the fastText models attain the highest similarity scores for the domain-specific terms for models in both pre-trained group and the self-trained group, which could indicate that this model is the best choice for the entity extraction task. However, the accuracy of entities extracted using these model is the lowest among all models. Comparing to fastText models, both the pre-trained and self-trained word2vec CBOW models achieve stable and better results. They also achieve better results than the word2vec SG models and the GloVe models, indicating word2vec CBOW models are a good choice for the entity extraction task. We believe that the reason why word2vec CBOW models outperform the fastText, GloVe, and word2vec SG models is that word2vec CBOW models are better at capturing contextual information while fastText models are more capable of capturing subword information. GloVe models are much better at capturing the word-word co-occurrence feature, whereas the word2vec SG models are said to give a better performance when the dataset is small[17]. For the self-trained word2vec word embedding models, the SG and CBOW model which is merely trained on our own dataset shows that the CBOW model achieves a better performance in terms of its cosine similarity score since more of the domain-specific characteristics can be captured when models are trained on a domain-specific corpus. Meanwhile, the self-trained word2vec CBOW model, which is trained on wiki12 and our own dataset, with the window size set to 2, achieves the best performance in terms of the cosine similarity score and the one trained on our own dataset, with the window size set as 2, achieves the highest accuracy, indicating that a smaller window size with CBOW models could be better at capturing features needed for our task. Through all these comparisons, we could conclude that self-trained word2vec models can achieve a better performance for the entity extraction task when the accuracy score is considered as the evaluation metric. The best performance is still given by the pre-trained GoogleNews-vectors-negative300 model, however, it is believed that the performance of the word2vec models will be improved if a larger Wikipedia corpus is used in self-trained models.

5.2 Process Extraction

For the process extraction task, the results for processes extracted with the baseline method and the distributional semantics-based method are given in this section and the discussion will mainly focus on: 1) comparing the performance of pre-trained models and self-trained models for the distributional semantics based method; 2) comparing the baseline method and the distributional semantics method in terms of precision, recall and F score. The precision, recall and F score are calculated as described in equation 4, 5, 6, 7, 15.

5.2.1 Results. We ran the various process extraction methods on all abstracts in our corpus, resulting in a table with abstract numbers and extracted triplets per abstract for each method. Table 3 shows the number of abstracts with triplets extracted using distributional semantics-based method and baseline method for the process extraction task. The baseline method could only extract biorefinery processes from 22 abstracts (0.22% of the corpus), whereas 3,255 abstracts (33.11% of the corpus) with triplets can be extracted with the fasttext-wiki-news-subwords-300 model, which seems to achieve the best performance among the pre-trained models. Except for

Model Name	bioethanol	gasification	man
fasttext-wiki-news-subwords-300	**0.78**	**0.47**	**0.20**
glove-wiki-gigaword-300	0.62	**0.41**	-0.08
glove-wiki-gigaword-100	0.48	0.30	0.01
GoogleNews-vectors-negative300	**0.75**	**0.45**	0.05
SG-OD-win2-300	0.66	0.23	0.00
CBOW-OD-win2-300	**0.72**	0.12	0.06
CBOW-Wiki1-OD-win2-300	0.73	0.19	-0.11
CBOW-Wiki12-OD-win10-300	0.75	0.26	-0.09
CBOW-Wiki12-OD-win5-300	0.78	0.30	-0.13
CBOW-Wiki12-OD-win2-300	**0.79**	**0.39**	-0.03
fastText-Wiki12-OD-300	**0.96**	**0.47**	-0.05

Table 1: Cosine similarity between 'ethanol' and other terms for different word embedding models.

Word Embedding Model	**Accuracy (%)**
GoogleNews-vectors-negative300	**76.67**
glove-wiki-gigaword-300	56.67
glove-wiki-gigaword-100	53.33
fasttext-wiki-news-subwords-300	33.33
fastText-Wiki12-OD-300	29.68
CBOW-Wiki12-OD-win2-300	70.12
CBOW-OD-win2-300	**73.33**
SG-OD-win2-300	33.33

Table 2: Accuracy for entity list expanded with different word embedding models

the pre-trained fasttext-wiki-news-subwords-300 model, triplets can be extracted from approximately 9% of the corpus with the GoogleNews-vectors-negative300 model, which is relatively high compared to the GloVe models (0.38% for glove-wiki-gigaword-300 and 3.64% for glove-wiki-gigaword-100). In terms of the self-trained models, the self-trained fastText model again achieves the best performance with triplets able to be extracted from 9.05% of the abstracts in the corpus, whereas triplets could only be extracted from 2.92%, 2.11% and 2.32% of the abstracts in the corpus for self-trained word2vec models.

Though it can be observed that the methods using fastText models (both pre-trained and self-trained) achieve rather good performances in the process extraction task when only considering the number of abstracts from which the triplets could be extracted, the quality of the extracted process (triplets) still needs to be examined, since the quality of extracted entities is not guaranteed. Therefore, it is necessary to validate the extracted triplets and then evaluate the models by comparing the results of all methods. We choose the models which give better performances on the entity extraction task and return a larger number of abstracts where triplets can be extracted. This intuitively means that there is a higher probability of getting the correctly extracted triplets with these models. The models are listed in Table 5. The reason why fastText models are not used in our experiments is that the accuracy of entities extracted using the fastText models is quite low and the accuracy of the extracted entities is a crucial aspect which could have an impact on process extraction results. The precision and recall scores are computed for the baseline method and the distributional semantics-based method with several significant word embedding models. The results are shown in Table 5. Both the macro-average precision and the micro-average precision for the baseline method are 0.5, whereas the macro average precision scores and the micro average precision scores for the distributional semantics-based method are roughly the same, not exceeding 0.5, for all three word embedding models. The word embedding model trained on our own dataset, with the window size set as two, achieves the best performance in our experiments using the distributional semantics method in terms of precision. When it comes to the recall of all methods we tried in our experiments, it is found that the distributional semantics-based method, using the self-trained CBOW-Wiki12-OD-win2-300 model, achieves the best performance and get a score of 0.5098 and 0.5556 for macro average recall and micro average recall, respectively. Meanwhile, the baseline model and the distributional semantics-based method using the CBOW-OD-win2-300 model, only get a score of 0.2319 and 0.125 for each of them while the distributional semantics-based method using the GoogleNews-vectors-negative300 model gets a score of 0.3056 in terms of the macro average recall. Furthermore, the micro average recall score for the baseline method is 0.2188 while it is 0.3431 for the distributional semantics-based method with the CBOW-OD-win2-300 model and the GoogleNews-vectors-negative300 model. Comparing the F score for all these models, the CBOW-Wiki12-OD-win2-300 model achieves the best performance since it gets the highest F score. In order to get an intuitive idea of how the precision and recall score changes for all experimented models, a scatterplot of the precision and recall scores is shown in Fig.5. The value maintains the same for each of the curved lines and the lines at the right side has a higher value than the curved line which is on the left side (The dots in the plot, baseline(ma), baseline(mi), google(ma), google(mi), cbow(ma), cbow(mi), cbow_wiki(ma) and cbow_wiki(mi), are the dots for the corresponding precision and recall scores for the baseline method, the GoogleNews-vectors-negative300 model, the CBOW-OD-win2-300 model, and the CBOW-Wiki12-OD-win2-300 model, respectively.). Therefore, it is easy to conclude from Figure. 5 that the CBOW-Wiki12-OD-win2-300 model works the best in terms of F-score.

Process extraction method	Number of articles
Baseline method	**22 / 9,831 = 0.22** %
GoogleNews-vectors-negative300	**879 / 9,831 = 8.94%**
glove-wiki-gigaword-300	37 / 9,831 = 0.38%
glove-wiki-gigaword-100	358 / 9,831 = 3.64%
fasttext-wiki-news-subwords-300	**3,255 / 9,831 = 33.11%**
fastText-Wiki12-OD-300	**890 / 9831 = 9.05%**
CBOW-Wiki12-OD-win2-300	288 / 9,831 = 2.92%
CBOW-OD-win2-300	207 / 9831 = 2.11%
SG-OD-win2-300	229 / 9,831 = 2.32%

Table 3: The number of abstracts contains triplets using distributional semantics-based method and the baseline method

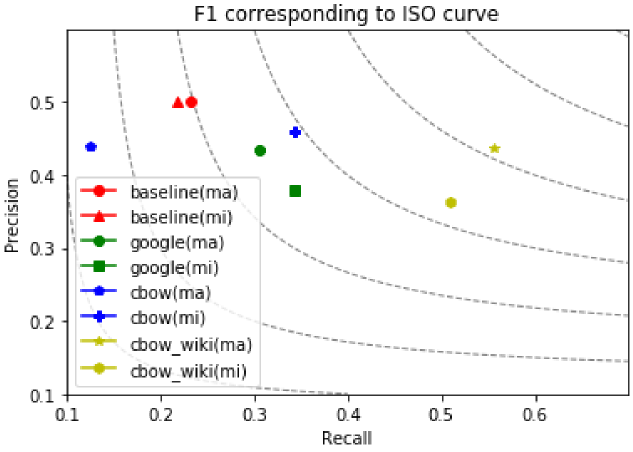

Figure 5: Scatterplot of the micro and macro precision and recall for experiments listed in Table 5. The dashed curved lines are F score ISO-curves: F score remains the same for all precision/recall combinations for the same curve line. Note that the axis starts from 0.1 and end at 0.5 for the Y-axis and 0.6 for the X-axis.(ma and mi is short for macro and micro, respectively)

5.2.2 Discussion – Self-trained Models and Pre-trained Models for Process Extraction Task. As shown in Table 3, 5 and Fig. 5, although pre-trained models can extract triplets from more abstracts, the precision and recall scores for self-trained models are higher, indicating that the self-trained models perform better in the process extraction task. We attribute this to the low accuracy of extracted entities, and the incompleteness of the vocabulary for the pre-trained models. The pre-trained models get a lower accuracy for the entity extraction task since more general topics are covered in the corpus used for model training, while the self-trained models are more capable of capturing domain-specific semantic meaning, since they are trained on the domain-specific corpus. As the accuracy of the extracted entities weighs heavily upon the process extraction task, the self-trained models are giving better performances in the process extraction task. Meanwhile, in terms of the completeness of the vocabulary for each model, the self-trained models have the advantage that vector representations for all domain-specific terms are available, since the models are trained on a domain-specific corpus. Therefore, it is more likely that the recall scores for the self-trained models are higher. As shown in Fig.5, it is found that the self-trained models tend to get higher recall scores while precision scores remain similar to the pre-trained models. However, it is not true when we compare the results for the GoogleNews-vectors-negative300 model and the CBOW-OD-win2-300 model as the self-trained CBOW-OD-win2-300 model gets a lower recall compared to the pre-trained GoogleNews-vectors-negative300 model. The intuitive reason we deduced for why the CBOW-Wiki12-OD-win2-300 model obtains higher recall than the CBOW-OD-win2-300 model is that the CBOW-Wiki12-OD-win2-300 model is trained on a corpus which includes not only all domain-specific articles, but also the articles in generic topics, which helps in distinguishing the semantics for domain-specific terms from general domain terms. Meanwhile, this reason also explains why the CBOW-OD-win2-300 model achieves a higher precision score than the CBOW-Wiki12-OD-win2-300 model. Limited only in this work, we could summarize from the above discussion that self-trained models can give better performance than the pre-trained models for the process extraction task.

5.2.3 Discussion – Comparison between the Distributional Semantics-based Method and the Baseline Method . Table 3 shows that the number of abstracts where triplets are extracted using the baseline method is 22, occupying 0.22% of the corpus, while the distributional semantics-based method with different models, not including the glove-wiki-gigaword-300 model, finds triplets in many more abstracts (at least 10 times more). What is more, the baseline method achieves the highest score at 0.5 for both the micro average precision and macro average precision and gets pretty low recall scores, while the distributional semantics-based method with the CBOW-Wiki12-OD-win2-300 model gets higher recall scores and lower precision scores. However, the macro average recall score for the CBOW-OD-win2-300 model is the lowest, while the distributional semantics-based method is being used. The reason for this could be attributed to the fact that merely the domain-specific articles are included in the corpus used for training the model, thereby weakening the model's ability to distinguish between domain-specific terms and other terms. Therefore, it is fair to conclude that the baseline method achieves higher precision scores than the distributional semantics-based method with word embedding models while the distributional semantics-based method with word embedding models trained on corpora which contain both domain-specific terms and terms from general topics achieves higher recall scores.

Micro and macro precision, recall and F score are computed here as evaluation metrics. Both macro and micro scores are computed here since the scores can be computed, in the macro calculation method, considering the difference among all abstracts whereas for the micro method, all triplets extracted for all abstracts are evaluated as a whole. Thus, macro evaluation metrics are more suitable for situations where single document difference is considered while micro evaluation method is more of a choice when the documents are evaluated as a whole set.

6 CONCLUSION AND FUTURE WORK

6.1 Conclusion

This thesis gives the definition of the simplified process metamodel for the biorefining technology and describes a text mining pipeline for extracting the processes from scientific literature. A baseline method (dictionary look-up) and a distributional semantics-based method with different word embedding models expanding the seed list is proposed to extract biorefinery processes from abstracts following the text mining pipeline. Experiments are designed and implemented to compare the performance of the baseline method with the distributional semantics-based method with different word embedding models. The main findings of our work are:

- In terms of the entity extraction task, pre-trained and self-trained word2vec word embedding models obtain better performances than other models when considering both accuracy and the cosine similarity metrics, thus word2vec models seem to be more suitable for an entity extraction task. Meanwhile, the result rejects hypothesis 1 since the pre-trained GoogleNews-vectors-negative300 model gets better results than the self-trained word2vec models. What is more, it should be noted that although fastText models achieve better results when considering only the similarity score, the accuracy is pretty low when using this model, and therefore fastText models are not suitable for the entity extraction task when compared with word2vec models. Furthermore, it is found that self-trained models with a smaller window size give higher accuracy and cosine similarity scores for similar terms. What is more, self-trained word2vec models trained with the CBOW model perform on entity extraction tasks.
- For the process extraction task with the distributional semantics-based method, comparing different word embedding models, it is found that the self-trained word embedding model, trained on merely a domain-specific corpus, can obtain a better performance in terms of precision while the self-trained model which is trained on corpora containing domain-specific articles and articles for generic topics can get a higher recall score. However, the results shows that self-trained models do not outperform the pre-trained GoogleNews-vectors-negative300 model, which rejects the hypothesis 2.
- In this process extraction task, the baseline method can be used to extract only a limited number of processes from a limited number of abstracts since only a small number of entities are available for constructing triplets. The distributional semantics-based method is able to extract more biorefinery processes from abstracts since more entities are available when the word embedding is used to find similar terms. However, the precision of the baseline method is higher than the distributional semantics-based method since all terms in the seed list are valid while some of the terms extracted using the distributional semantics-based method could be irrelevant and thus decrease the number of correctly detected triplets. The distributional semantics-based method with the word embedding model trained on the corpus containing domain-specific articles and articles on generic topics tends to give higher recall and lower accuracy compared with the baseline method, since the number of entities in the seed list is limited. It is found most of the experiments with distributional semantics-based method get a higher recall and lower precision comparing to the baseline method, we could conclude that the hypothesis 3 is mostly accepted.

Although the baseline method can be used to extract processes from biorefinery articles, it is particularly time-consuming and laborious for experts to annotate the large volume of data. The results in this work show that distributional semantics-based methods are capable of extracting biorefinery processes from abstracts starting with little training data (seed list) and even outperforms the baseline method in some cases in terms of recall, such as the experiment with the CBOW-Wiki12-OD-win2-300 model.

6.2 Limitations and Recommendation for Future Work

The limitations and future work of our study are listed as follows:

- The processes are extracted on single sentence level. However, the real situation is that (1) one process might be described with more than one sentence; (2) processes can be described in not only the abstracts, but also other part of the articles.

  With regard to these situations, a potential solution is to model the process extraction task as a neural machine translation task using sequence2sequence models[7] in which the texts (could be only the abstracts or other parts of the articles) are taken as the input and the processes are the output. However, worth noting for future research would be that this method requires a lot of training data. The method is not restricted to the suggested one, and any method which could be used to extract processes across sentences or in a paragraph is worth experimenting with.
- While inputs and outputs from a biorefinery can be clearly defined, it is difficult to identify the single most important technology if it involves several; similarly, it can be difficult to name the single most important input and output if a biorefinerey uses a mix of biomass sources and produces several products. In this work, the processes are seen as correctly identified if the input, technology and output matches. The relevance of the process for that article is not considered.

  To solve the problems described in the above limitations, a relevance scoring method for filtering out the relevant processes is needed. It could also be useful to implement active-learning algorithms, where experts are involved in identifying the validity of the processes while extracting processes.
- Although the dataset used for process extraction is already filtered using the keywords when retrieving it via Elsevier's API, it is found in our experiments that some of the articles are irrelevant for process extraction, since some articles are reviews, while others describe model integration tools, etc. The irrelevant articles would probably lead to worse performance of the process extraction algorithms. It is meaningless to have the irrelevant articles for process extraction task since the irrelevant articles do not contain processes

[7] https://medium.com/analytics-vidhya/a-must-read-nlp-tutorial-on-neural-machine-translation-the-technique-powering-google-translate-c5c8d97d7587

that we need. Meanwhile, it is difficult to define hard criteria of which articles are the ones that should be considered and the others that do not describe an original technology or only a part of a biorefinery, or biorefineries in general.

In order to improve the performance of the process extraction task, it is practical to apply a document classifier to filter out articles which only focus on describing the biorefinery processes and thus reduce the noises caused by irrelevant articles. Human intervention on checking the validity of the filtered documents could also be considered.

- Another limitation of our work is that the methods we proposed in this work are not able to determine the order of the input and the output in a process, since the input can sometimes be the output of another process and vice-versa, the output can also be an input in different processes. Similar to the chemical reaction extraction or protein-protein interaction extraction task where recognizing the order of entities in the relation instance is challenging. The order of the input and output is also a challenge for the process extraction task [25] [29].

  This limitation might be addressed by asking experts to find out and define which patterns are used to describe the inputs and outputs and then classify the entities into inputs and outputs according to the pattern it belongs to. However, the result is not guaranteed, and it is also time-consuming to find out all the patterns. Furthermore, some of the patterns might be used for describing both inputs and outputs. What is more, this would also be of limited use since some of the terms are ambiguous, such as 'residual biomass', which can refers to lots of inputs and has different meaning in different context.
- The quality of the entities extracted in the distributional semantics-based method in this work can be improved since it affects the performance of the process extraction.

  The quality of the extracted entities could be improved by either manually selecting the relevant ones from extracted entities or designing an algorithm to automatically filter our relevant entities.

Except for future work suggested according to the limitations of this work, we could also manually construct an ontology and use our baseline method to extract the process, which is expected to give higher accuracy but involve more intervention from experts. A methodology which is capable of iteratively building a semantic model of a field, involving text mining and crowd sourced vetting (among experts) can be worth a try. Furthermore, the process model defined in this work is merely aiming at extracting inputs and outputs of a process, however, a process is far more than this, a more specific model in which the quantity of the inputs and outputs, the agency where the process happens and catalyst or enzymes included in processes, etc. can be defined in future work. What is more, the abstracts for generating the seed list in this work is randomly selected, it is suggested that generated the seed list from abstracts that are intentionally selected which contain certain kind of technologies or inputs and outputs. Last but not least, word embedding models trained on a larger corpus is also one aspect which can be studied in future work since the available Wikipedia corpus is far larger than the one used in this work, and patents or other forms of domain-specific articles could be included in the corpus as well.

REFERENCES

[1] W.M.P. Aalst, van der. 2011. *Process mining : discovery, conformance and enhancement of business processes.* Springer, Germany. https://doi.org/10.1007/978-3-642-19345-3

[2] Nguyen Bach and Sameer Badaskar. 2011. A Review of Relation Extraction. (05 2011).

[3] bioeconomy 2019. Global importance of bioeconomy. Retrieved May 29, 2019 from https://biooekonomie.uni-hohenheim.de/en/bioeconomy-global

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* 5 (07 2016). https://doi.org/10.1162/tacl_a_00051

[5] Quoc-Chinh Bui. 2012. Relation extraction methods for biomedical literature. *Structure* (01 2012).

[6] Chris Davis, Graham Aid, and Ben Zhu. 2017. Secondary Resources in the Bio-Based Economy: A Computer Assisted Survey of Value Pathways in Academic Literature. *Waste and Biomass Valorization* (06 2017), 1–18. https://doi.org/10.1007/s12649-017-9975-0

[7] J Ding, D Berleant, D Nettleton, and Eve Wurtele. 2002. Mining MEDLINE: abstracts, sentences, or phrases? *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* 7 (02 2002), 326–37. https://doi.org/10.1142/9789812799623_0031

[8] Mohamed Firdhous. 2010. Automating Legal Research through Data Mining. *CoRR* abs/1211.1861 (2010).

[9] Katrin Fundel-Clemens, Robert KÃijffner, and Ralf Zimmer. 2007. RelEx - Relation extraction using dependency parse trees. *Bioinformatics (Oxford, England)* 23 (03 2007), 365–71. https://doi.org/10.1093/bioinformatics/btl616

[10] Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, Franciska de Jong, and Emiel Caron. 2016. A Survey of event extraction methods from text for decision support systems. *Decision Support Systems* 85 (02 2016). https://doi.org/10.1016/j.dss.2016.02.006

[11] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Association for Computational Linguistics, Melbourne, Australia, 328–339. https://www.aclweb.org/anthology/P18-1031

[12] Hui-Huang Hsu, Chuan-Yu Chang, and Ching-Hsien Hsu. 2017. *Big Data Analytics for Sensor-Network Collected Intelligence* (1st ed.). Academic Press, Inc., Orlando, FL, USA.

[13] Jung-Jae Kim, Zhuo Zhang, Jong Park, and See-Kiong Ng. 2006. BioContrasts: Extracting and exploiting protein-protein contrastive relations from biomedical literature. *Bioinformatics (Oxford, England)* 22 (04 2006), 597–605. https://doi.org/10.1093/bioinformatics/btk016

[14] Linsey Koo, Nikolaos Trokanas, and F Cecelja. 2017. A semantic framework for enabling model integration for biorefining. *Computers & Chemical Engineering* 100 (02 2017), 219–231. https://doi.org/10.1016/j.compchemeng.2017.02.004

[15] Haibin Liu, Ravikumar Elayavilli, and Karin Verspoor. 2011. From Graphs to Events: A Subgraph Matching Approach for Information Extraction from Biomedical Text. *Proceedings of BioNLP Shared Task 2011 Workshop* (06 2011), 164–172.

[16] Kes McCormick and Niina Kautto. 2013. The Bioeconomy in Europe: An Overview. *Sustainability* 5 (06 2013), 2589–2608. https://doi.org/10.3390/su5062589

[17] Tomas Mikolov, Kai Chen, G.s Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *Proceedings of Workshop at ICLR* 2013 (01 2013).

[18] Sheshera Mysore, Edward Kim, Emma Strubell, Ao Liu, Haw-Shiuan Chang, Srikrishna Kompella, Kevin Huang, Andrew Mccallum, and Elsa Olivetti. 2017. Automatically Extracting Action Graphs from Materials Science Synthesis Procedures. (11 2017).

[19] Philip Nuss. 2014. Life Cycle Assessment Handbook: A Guide for Environmentally Sustainable Products, edited by Mary Ann Curran. Hoboken, NJ, USA: John Wiley & Sons, Inc., and Salem, MA, USA: Scrivener Publishing LLC, 2012, 611 pp., ISBN 9781118099728, 199.00$(paper)$,159.99 (e-book). *Journal of Industrial Ecology* 19 (12 2014). https://doi.org/10.1111/jiec.12217

[20] Sachin Pawar, Girish Palshikar, and Pushpak Bhattacharyya. 2017. Relation Extraction : A Survey. (12 2017).

[21] Jeffrey Pennington, Richard Socher, and Christoper Manning. 2014. Glove: Global Vectors for Word Representation. *EMNLP* 14, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[22] Sampo Pyysalo, Antti Airola, Juho Heimonen, Jari BjÃűrne, Filip Ginter, and Tapio Salakoski. 2008. Comparative analysis of five protein-protein interaction corpora. *BMC bioinformatics* 9 Suppl 3 (02 2008), S6. https://doi.org/10.1186/1471-2105-9-S3-S6

[23] Fabio Rinaldi, Gerold Schneider, Kaarel Kaljurand, Michael Hess, Christos Andronis, Ourania Konstanti, and Andreas Persidis. 2007. Mining of relations between proteins over biomedical scientific literature using a deep-linguistic approach. *Artificial intelligence in medicine* 39 (03 2007), 127–36. https://doi.org/10.1016/j.artmed.2006.08.005

[24] J RockstrÃűm, W Steffen, Kevin Noone, ÃĚsa Persson, F Stuart Chapin III, E.F. Lambin, T M. Lenton, M Scheffer, Carl Folke, Hans Schellnhuber, BjÃűrn Nykvist, Cynthia de Wit, Terence Hughes, Sander Van der Leeuw, H Rodhe, Sverker SÃűrlin, P K. Snyder, Robert Costanza, Uno Svedin, and J A. Foley. 2013. A safe operating space for humanity. *Nature, v.461, 472-475 (2009)* 46 (01 2013).

[25] Jhuma Sadhukhan, Kok Siew Ng, and Elias Martinez. 2014. *Biorefineries and Chemical Processes: Design, Integration and Sustainability Analysis.* https://doi.org/10.1002/9781118698129.ch15

[26] Isabel Segura-Bedmar, VÃŋctor SuÃąrez-Paniagua, and Paloma Martinez. 2015. Exploring Word Embedding for Drug Name Recognition. 64–72. https://doi.org/10.18653/v1/W15-2608

[27] Miran Seok, Hye-Jeong Song, Chan Park, Jong-Dae Kim, and Yu-Seop Kim. 2015. Comparison of NER Performance Using Word Embedding. 784–788. https://doi.org/10.14257/astl.2015.120.154

[28] Pravin Shinde and Sharvari Govilkar. 2015. A Systematic study of Text Mining Techniques. *International Journal on Natural Language Computing* 4 (08 2015), 54–62. https://doi.org/10.5121/ijnlc.2015.4405

[29] Rune SÃętre, Kenji Sagae, and Jun'ichi Tsujii. 2007. Syntactic Features for Protein-Protein Interaction Extraction. *Proceedings of the International Symposium on Languages in Biology and Medicine (LBM short oral presentations)* 6.

[30] Ramzan Talib, Muhammad Kashif, Shaeela Ayesha, and Fakeeha Fatima. 2016. Text Mining: Techniques, Applications and Issues. *International Journal of Advanced Computer Science and Applications* 7 (11 2016). https://doi.org/10.14569/IJACSA.2016.071153

[31] Ah-Hwee Tan. 2002. Text Mining: Promises And Challenges. (02 2002).

[32] Nikolaos Trokanas, Madeleine Bussemaker, Eirini Velliou, Hella Tokos, and F Cecelja. 2015. BiOnto: An Ontology for Biomass and Biorefining Technologies. *Computer Aided Chemical Engineering* 37 (06 2015), 959. https://doi.org/10.1016/B978-0-444-63577-8.50005-X

[33] Miguel Vazquez, Martin Krallinger, Florian Leitner, and Alfonso Valencia. 2011. Text Mining for Drugs and Chemical Compounds: Methods, Tools and Applications. 30, 6âĂŘ7 (2011), 506–519. https://doi.org/10.1002/minf.201100005 Exported from https://app.dimensions.ai on 2019/05/27.

[34] Hei-Chia Wang, Yi-Hsiu Chen, Hung-Yu Kao, and Shaw-Jenq Tsai. 2011. Inference of transcriptional regulatory network by bootstrapping patterns. *Bioinformatics (Oxford, England)* 27 (03 2011), 1422–8. https://doi.org/10.1093/bioinformatics/btr155

[35] Sholom Weiss, N Indurkhya, T Zhang, and F J. Damerau. 2004. *Text Mining: Predictive Methods for Analyzing Unstructured Information.* https://doi.org/10.1007/978-0-387-34555-0

[36] Kaixian Yu, Pei-Yau Lung, Tingting Zhao, Peixiang Zhao, Yan-Yuan Tseng, and Jinfeng Zhang. 2018. Automatic extraction of protein-protein interactions using grammatical relationship graph. *BMC Medical Informatics and Decision Making* 18 (07 2018). https://doi.org/10.1186/s12911-018-0628-4

A TABLES

A.1 Word embedding models used in this work – See Table 4.

B SUPPLEMENT CONTENT

B.1 Full description for libraries

- Elementtree v1.2.6-20050316: For parsing and creating XML data. This library is used in the preprocessing step for processing the retrieved XML files.
- Regex 2019.04.14: Regular expression module which offers regular expression matching operations. It is mainly used for preprocessing in this work.
- NLTK (Natural Language Toolkit) v3.4.1: A python package for natural language processing. Mainly involved in preprocessing, named entity recognition (NER) and relation extraction (RE) tasks.
- Gensim v3.4.0: A free Python library, mainly consists of unsupervised algorithms, which could be used for automatically extracting semantic topics from documents with high efficiency compared to manual operations. In this work, pretrained word embedding models generated with Word2Vec, FastText, GloVe and self-trained Word2Vec models are used and compared in named entity recognition (NER) task. This library is used in named entity recognition (NER) task to expand the entity list.
- Numpy v1.15.14: A fundamental package in python which is usually used for scientific computing to improve the efficiency of storing and processing large matrices. In this work, it is used for improving the efficiency of storing and operating lists and matrices.
- Scikit-learn v0.20.1: A free library for machine learning which could be easily reached by programming in Python. This library is used for creating some machine learning models.

B.2 Definition of evaluation metrics

B.2.1 Similarity Score. The method for calculating cosine similarity score between two vectors (vector A and vector B) is shown in equation 2.

$$Sim(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}||\ ||\mathbf{B}||} \tag{2}$$

B.2.2 Accuracy. Accuracy, as described in equation 3 is defined as the number of relevant instances (n_{relEnt}) divided by the total number of instances ($n_{totalEnt}$) recognized using word embedding models.

$$Acc = \frac{n_{relEnt}}{n_{totalEnt}} \tag{3}$$

B.2.3 Precision. Micro average precision (see equation 4) is defined as the total number of correctly extracted triplets (n_{relTri}) divided by the total number of extracted triplets ($n_{totalTri}$). Macro average precision is calculated by first summing up the total number of correctly extracted triplets for each abstract ($n_{relTriAbs}$) divided by the total number of extracted triplets for each abstract ($n_{totalTriAbs}$), and then the sum is divided by the number of abstracts (n_{abs}) used for extracting triplets, see equation 5.

$$Micro\_Ave\_Precision = \frac{n_{relTri}}{n_{totalTri}} \tag{4}$$

$$Macro\_Ave\_Precision = \frac{\sum \frac{n_{relTriAbs}}{n_{totalTriAbs}}}{n_{abs}} \tag{5}$$

B.2.4 Recall. Micro average recall is defined as the number of correctly extracted triplets (n_{relTri}) divided by the total number of correct triplets ($n_{totalRelTri}$) as shown in equation 6 and the macro average recall is calculated as described in equation 7 by first adding up all the recall scores for each abstract and then take the average. $n_{trueTriAbs}$ represents the number of relevant triplets in each abstract which are manually identified and serve as the ground-truth triplets for evaluation.

$$Micro\_Ave\_Recall = \frac{n_{relTri}}{n_{totalRelTri}} \tag{6}$$

$$Macro\_Ave\_Recall = \frac{\sum \frac{n_{relTriAbs}}{n_{trueTriAbs}}}{n_{absTrue}} \tag{7}$$

	Model Architecture	Model Name	Description
Pre-trained	CBOW	fasttext-wiki-news-subwords-300	1 million 300-dimensional vectors trained on Wikipedia 2017, UMBC webbase corpus and statmt.org news dataset with window size set to 5
	GloVe	glove-wiki-gigaword-300	400,000 unique 300-dimensional vectors trained on Wikipedia 2014 and Gigaword 5
	GloVe	glove-wiki-gigaword-100	400,000 unique 100-dimensional vectors trained on Wikipedia 2014 and Gigaword 5
	CBOW	GoogleNews-vectors-negative300	3 million 300-dimensional vectors trained on part of Google News dataset with window size set to 5
Self-trained	SG	SG-OD-win2-300	285,684 unique 300-dimensional vectors trained on OD with window size set to 2
	CBOW	CBOW-OD-win2-300	285,684 unique 300-dimensional vectors trained on OD with window size set to 2
		CBOW-Wiki1-OD-win2-300	760,928 unique 300-dimensional vectors trained on wiki1 + OD with window size set to 2
		CBOW-Wiki12-OD-win10-300	956,214 unique 300-dimensional vectors trained on wiki12 + OD with window size set to 10
		CBOW-Wiki12-OD-win5-300	956,214 unique 300-dimensional vectors trained on wiki12 + OD with window size set to 5
		CBOW-Wiki12-OD-win2-300	956,214 unique 300-dimensional vectors trained on wiki12 + OD with window size set to 2
		fastText-Wiki12-OD-300	956,214 unique 300-dimensional vectors trained on wiki12 + OD with window size set to 2

Table 4: Word embedding models used in this work(OD means own dataset which is retrieved via Elsevier's API, wiki1 means the wiki dump1 dataset downloaded from wiki dump, while wiki12 means wiki1 and wiki2 dump that downloaded from wiki dump.)

Process extraction method	Micro_Precsion	Macro_Precision	Micro_Recall	Macro_Recall	Micro_F	Macro_F
Baseline method	**14 / 28 = 0.5**	**0.5**	14 / 64 = 0.2188	0.2319	0.1521	0.1584
GoogleNews-vectors-negative300	35 / 92 = 0.3804	0.435	35 / 102 = 0.3431	0.3056	0.1804	0.1795
CBOW-OD-win2-300	**35 / 76 = 0.4605**	**0.44**	35 / 102 = 0.3431	0.125	0.1966	0.0973
CBOW-Wiki12-OD-win2-300	52 / 143 = 0.3636	0.4360	**52 / 102 = 0.5098**	**0.5556**	0.2122	0.2443

Table 5: Precision, recall, and F-score for process extracted using different models.

C DOCUMENT CLASSIFICATION

C.1 Document Representation Methods

C.1.1 Bag of Words (BOW) Model. The BOW model, commonly used in document classification methods, uses the occurrence of each word as the feature to train a classifier. It is a sparse vector representation of text where each document is represented as a vector consists of occurrence count for each word in the corpus. With this method, we assume that higher values of a word indicates greater importance of the word.

C.1.2 TF-IDF. TF-IDF, an occurrence-based numeric representation of text where TF stands for term frequency and IDF for Inverse Document Frequency, assigns a high value for a given term if that term seldom occurs in other documents other than that particular document. The equations for calculating TF, IDF and TF-IDF are shown in equation 8, 9, 10 respectively.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}} \tag{8}$$

$$idf(i) = log\left(\frac{N}{df_i}\right) \tag{9}$$

$$tf - idf_i = tf_{i,j} \times idf(i) \tag{10}$$

For the above equations, $tf_{i,j}$ is the number of occurrences of term i in j document and df_i represents the number of documents including term i whereas N stands for the total number of documents in the document collection. A TF-IDF vector representation for each document can be computed by calculating TF-IDF score for each word in the document.

C.1.3 Word Embedding for Document Representation. Word embedding can be used as word representation, phrase representation and document representation. For words that has a vector representation in the word embedding model, the word can be represented by E(x), the word embedding for word x, such as E("biorefinery"). The word embedding representation for a document can be computed using equation 11.

$$E(D) = \frac{\sum_{i=1}^{n} E(w_i)}{n} \tag{11}$$

In equation 11, E(D) stands for the word embedding for a document D, n is the number of words in the document while E(w_i) represents the word embedding for i^{th} word in the document. The word embedding models used for document classification can be found in Table 4.

C.2 Document Classifiers

Since the collected dataset is composed of several article types, mainly from research articles, book chapters to review articles. Some of the articles seems less relevant for biorefinery process extraction since some of the articles are merely reviews, policy related, etc. Therefore, different classifiers were trained with different document representation methods described in last section for classifying documents.

C.2.1 Linear Support Vector Machine (SVM). With a training dataset of n points in the form of ($\vec{x}_1$, y_1), ... ($\vec{x}_i$, y_i), ... ($\vec{x}_n$, y_n), and y_i is either -1 or 1, which represents the class for point $\vec{x}_i$, the linear SVM model is used to find the "maximum-margin hyperplane" to separate the points $\vec{x}_i$ into 2 point sets, y_i = 1 or y_i = -1. It gives the maximized distances between hyperplane and the closest point from either point sets.

C.2.2 Logistic Regression (LG). Logistic regression is a typical method used for solving binary classification problems where the dependent variables are dichotomous in nature. The core of the logistic regression is the estimation of a multiple linear regression function, show in equation 12:

$$logit(p) = log(\frac{p(y=1)}{1-p(y=1)}) = \beta_0+\beta_1 x_1+...+\beta_i x_i+...+\beta_n x_n \tag{12}$$

y is a binary response variable and the parameters of the model are β_i coefficients.

C.2.3 ULMFiT. Universal Language Model Fine-tuning is a state-of-the-art transfer learning method proposed by Jeremy [11] which can be applied to any task in NLP. In this paper, a ULMFit model pretrained on English wikitext103 (103M tokens) is fine-tuned for custom dataset to better adapt to the linguistic properties of our own corpus. Then the fine-tuned language model is utilized in the text classification task.

C.2.4 Flair. Flair, a state-of-the-art NLP library, can be used in various NLP tasks, such as NER, text classification, etc. Different contextual and word embeddings can be combined to generate document embeddings for training a prediction model. The Flair with different word embedding combinations such as *GloVe*, *ELMo*, *word2vec*, etc. are studied in this work.

With all the classifiers listed above, given the document representation as input, we could get a prediction of the document whether it is relevant to process extraction or not. In this task, 122 abstracts from biorefinery articles are annotated by an expert in the field of biorefinery with nearly half of abstracts relevant for process extraction and half irrelevant.

C.3 Evaluation of Document Classification

C.3.1 Evaluation Metrics. In this document classification task, common evaluation metrics including precision, recall, f-score and accuracy are calculated to evaluate the performance of the classifiers. The definition of the evaluation metrics are listed as follows:

- Precision
  - The precision of a document classifier can be calculated as described in equation 13 where tp represents true positive predictions and fp stands for false positive predictions. True positive here means that the positive class is correctly predicted by the model where false positive means the positive class is incorrectly predicted by the model.

$$Precision = \frac{tp}{tp+fp} \tag{13}$$

- Recall
  - Recall for a document classifier is computed following the equation 14 where tp represents true positive predictions and fn stands for false negative predictions. Here the false negative means that the negative class is incorrectly predicted by the model.

$$Recall = \frac{tp}{tp+fn} \tag{14}$$

- F-score
  - F-score is the harmonic mean of precision and recall it is calculated as described in equation 15.

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{15}$$

- Accuracy
  - The accuracy is computed following the equation 16 where tn stands for true negative, indicating that the negative class is correctly predicted by the model.

$$Accuracy = \frac{tp+tn}{tp+tn+fp+fn} \tag{16}$$

C.3.2 Cross Validation (CV). Cross validation is used in our experiments since it is able to test how well the classification models are trained upon the dataset. We choose to work with k-fold (k = 5) and leave one out cross validation (LOO-CV). K is chosen considering the available instances since 5-fold cross validation means that 20% of the data is used for testing, which usually leads to pretty accurate result when the training dataset is small. The reason why LOO-CV is used here in order to compare with the k-fold cross validation and minimize the bias of the models.

- 5-fold CV
  - In this method, the given data set is split into 5 folds where each fold is used as a testing set at some point.
- Leave One Out Cross Validation (LOO-CV)
  - In this method, a single instance is used as the validation data and the rest of the instances are used as the training data, which is repeated until each instance in the data set is used once as the validation data.

C.4 Results and Discussions

The results for document classification task are shown in Table 6. Classifiers using word embedding as the document representation get rather low scores for all evaluation metrics while classifiers where document are represented using word count vectors achieve the best performance among the three document representing methods. With the intuition that no information is lost using word count vectors to represent documents while some of the terms are not included in word embedding models, the reason why word count vectors classifiers are achieving better performance can be explained. Meanwhile, it is observed that the scores for evaluation metrics with LOO-CV is higher than using 5-fold CV, indicating that it could be better to use LOO-CV when the training data set is small. However, though all Linear SVM in combination with LOO-CV achieves the best performance among all the experiments, the scores for all evaluation metrics are around 70%, which means there is still a high probability of classifying the articles into wrong categories. Thus, more document classification methods could be experiment in the future work to explore if there is a method to filter out all those articles which are relevant to process extraction task.

D CODE

The code for this project is available at https://github.com/zxlzhangxiaoling/master/tree/master/thesis/code

Document Representation	Classifier	Precision (%)	Recall (%)	F-score (%)	Accuracy (%)
TF-IDF vectors	LR (5-fold CV)	53.81	50.49	39.74	57.37
	LR (LOO-CV)	62.64	52.42	43.45	59.02
	Linear SVM (5-fold CV)	64.97	59.67	57.96	63.93
	Linear SVM (LOO-CV)	66.91	61.59	60.42	65.57
Word Count vectors	LR (5-fold CV)	66.39	66.00	66.11	67.21
	LR (LOO-CV)	67.25	66.95	67.06	68.03
	Linear SVM (5-fold CV)	68.85	68.10	67.66	67.81
	Linear SVM (LOO-CV)	**70.66**	**70.55**	**70.60**	**71.31**
GoogleNews-vectors-negative300	Linear SVM (LOO-CV)	47.62	42.81	45.08	46.23
Self-trained-CBOW-300 Word Embeddings	Linear SVM (LOO-CV)	39.41	36.13	37.70	37.25
Flair word embeddings	Flair	21.31	50.00	29.89	42.62
Word embeddings from Wikitext-103 language model	ULMFiT	22.00	50.00	30.56	44.00

Table 6: Document classification results for different classifiers, document representations and effectiveness testing method