Machine Learning for Software Requirements

Project Report Submitted by : **Harekrushna Mahapatra**

Department of Mathematics

National Institute of Technology, Rourkela

Under the guidance of : **Dr. Ir. Arpit Sharma**

Assistant Professor

Department of Electrical Engineering and Computer Science (EECS)

INDIAN INSTITUTE OF SCIENCE EDUCATION AND RESEARCH

BHOPAL - 462 066, Madhya Pradesh, INDIA

July 2022

ACKNOWLEDGEMENT

I want to extend a sincere and heartfelt obligation towards all the personages without whom the completion of the project was not possible. I express my profound gratitude and deep regard to Dr. Ir. Arpit Sharma, IISER Bhopal for his guidance, valuable feedback, and constant encouragement throughout the project. His valuable suggestions were of immense help. I sincerely acknowledge his constant support and guidance during the project.

I am also grateful to the Indian Institute of Science Education and Research, Bhopal, for allowing me to do this project and providing all the required facilities.

Bhopal - 462 066 **Harekrushna Mahapatra**

July 2022

ABSTRACT

The main aim of the project is to identify and detect the domain specific ambiguous words in natural language text by applying an NLP technique based on word embedding. Basically, a tool will be designed to analyze the context specific words in two different corpora. This will help to get an idea whether the common word is used in the same context in both the corpora. At last, to convince the user, the tool will be framing a sentence out of that word which is used in the same context in the respective corpus.

Keywords: Natural Language Processing, word embedding, requirements, machine learning, term extraction, nouns, semantic filter, ambiguity.

Contents

0.1 Introduction 5

0.2 Preliminaries 6

0.2.1 Word Embeddings 6

0.2.2 Word Similarity Computation 6

0.3 Approach 7

0.3.1 Corpus 7

0.3.2 Data Pre-processing 7

0.3.3 Word Frequency Count and Intersection 8

0.4 Results 9

0.4.1 CS *versus* CHE domain 10

0.4.2 CS *versus* MLE domain 10

0.4.3 CS *versus* MAR domain 11

0.4.4 CS *versus* PTE domain 13

0.4.5 CS *versus* ME domain 13

0.4.6 CS *versus* BME domain 14

0.4.7 CS *versus* CE domain 15

0.4.8 Limitations of the Approach 16

0.5 Conclusion 17

0.1 Introduction

Requirements are descriptions of how a system should behave. The quality of the requirements determines the overall quality of the software product. The various phases involved in software development are requirements, design, coding, testing, deploying. According to past research, around 60% of all errors in software development projects initiate during the requirements engineering phase. A major cause of poor quality requirements is that the stakeholders involved in the development phase have different interpretations of technical terms, and that is due to the ambiguity present in the natural language text.
For example, the word *cookie* in context of computer science (CS) means a small amount of data stored on user's machine by the web browser while it is a sweet biscuit for a food engineer. Similarly, the word *table* in context of computer science (CS) means a grid of rows and columns while it may also mean a furniture in different context.

This project focuses on extraction of text from requirements documents. Since 99% of all the relevant terms are noun phrases, we only focus on extracting them. We will be using neural word embedding technique for detecting the domain-specific technical terms from two different corpora (in PDF format). We will be using pre-trained *Word2Vec* model and *FastText* Python library, by the Facebook Research Team, for efficient learning of word representations and sentence classification.

Finally, we discuss the benefits and limitations of the above approach for analyzing requirements documents.

0.2 Preliminaries

0.2.1 Word Embeddings

Word embedding is a powerful approach for detecting domain-specific ambiguities in natural language text. It provides a dense representation of words in the form of numeric vectors which gives the semantic relationship of their meaning. Word embeddings are considered to be an improvement over the traditional bag-of-words model which results in very large and sparse word vectors. Out of various word embedding models, *FastText* model, developed by the Facebook Research Team, has been used in the project to estimate the ambiguity potential of typical computer science words when they are used in different domains. *FastText* is used over *Word2Vec* as *FastText* operates at a more granular level with character n-grams. This approach has been extended to estimate the variation of meaning of dominant shared terms in different domains by comparing the list of most similar words in each domain-specific model.

0.2.2 Word Similarity Computation

The Word2Vec model uses the cosine similarity to compute the semantic relationship of two different words in vector space.

Let us assume two word embedding vectors $\overrightarrow{w'}$ and $\overrightarrow{w''}$. The cosine angle between these two word embedding vectors is calculated using Equation (1).

$$\cos\left(\overrightarrow{w'}, \overrightarrow{w''}\right) = \frac{\overrightarrow{w'} \cdot \overrightarrow{w''}}{\left|\overrightarrow{w'}\right| \left|\overrightarrow{w''}\right|}$$

The range of similarity score is between 0 to 1 . The scores closer to 1 means that the words are semantically more similar and used in almost the same context. On the other hand, scores closer to 0 means that the words are less related to each other.

0.3 Approach

0.3.1 Corpus

The corpus is collected from CrowdRE dataset of various domains. The descriptive statistics of the collected data is provided in Table 1.

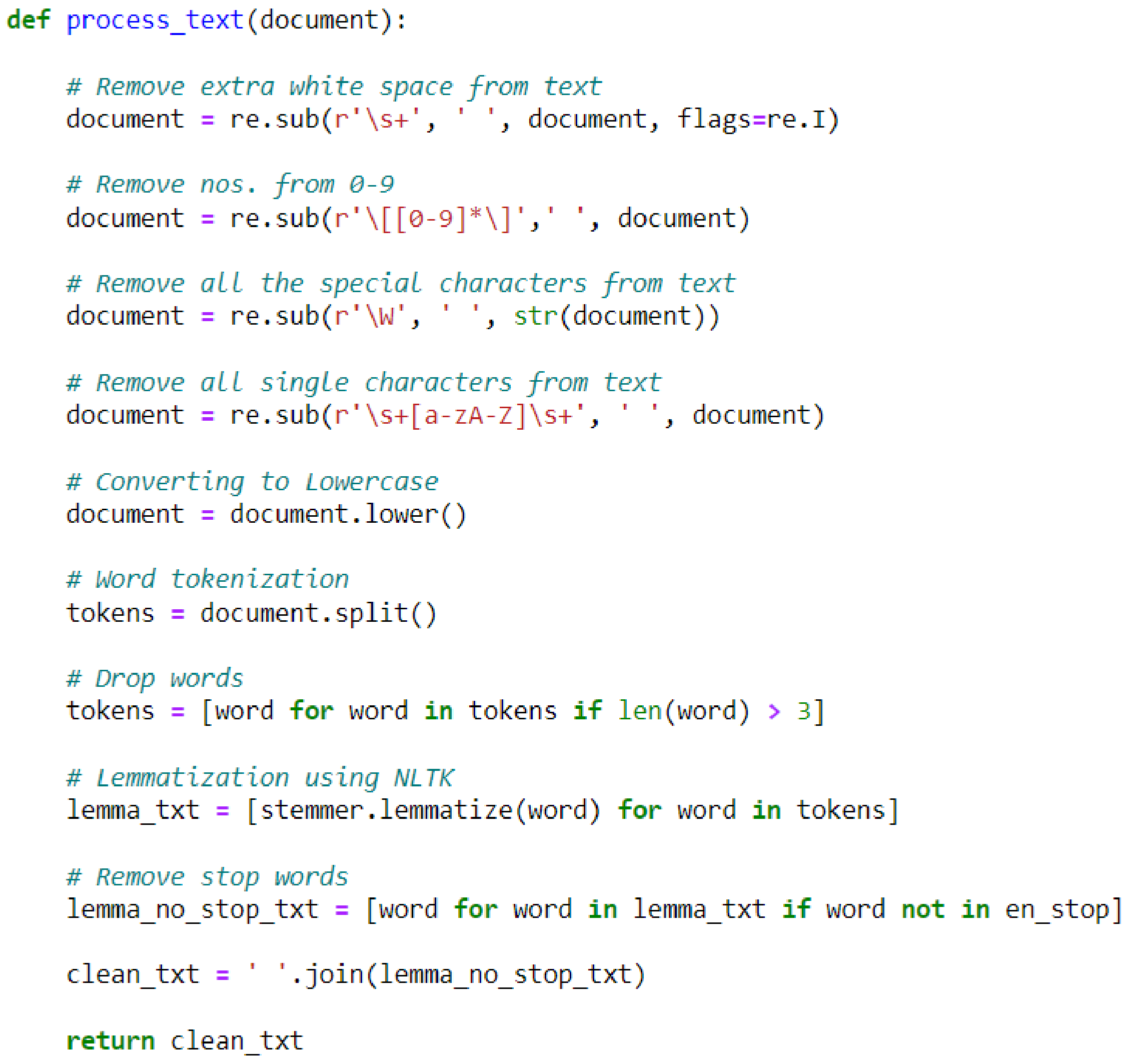
Category Name	Pages	Sentences	Words
Computer Science (CS)	34	$1,268$	$29,563$
Chemical Engineering (CHE)	61	$1,047$	$24,279$
Military Engineering (MLE)	48	555	$19,479$
Marine Engineering (MAR)	18	233	$8,056$
Petroleum Engineering (PTE)	15	207	$5,921$
Mechanical Engineering (ME)	14	290	$7,708$
Biomedical Engineering (BME)	14	206	$5,126$
Civil Engineering (CE)	12	294	$4,941$

0.3.2 Data Pre-processing

This step is useful for generating efficient word embedding vectors and preventing the vocabulary from becoming unnecessarily large. The textual data (sentences) of each corpus are broken into tokens of words (tokenization) followed by the cleaning of all special symbols, alpha-numeric words and punctuation marks (punctuation removal). Next, we convert all the words to lowercase (lowering of words) followed by the removal of noisy words defined for the English language (stopwords removal).

Finally, we lemmatize all the words. Lemmatization removes the inflectional endings and returns the base or dictionary form of a word, i.e., lemma.

Function to clean the text extracted from corpora (PDFs) :



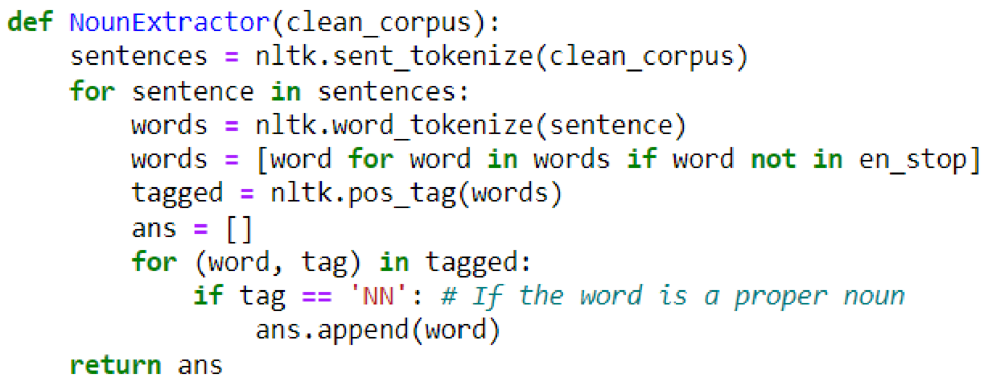
```
def process_text(document):

    # Remove extra white space from text
    document = re.sub(r'\s+', ' ', document, flags=re.I)

    # Remove nos. from 0-9
    document = re.sub(r'\[[0-9]*\]',' ', document)

    # Remove all the special characters from text
    document = re.sub(r'\W', ' ', str(document))

    # Remove all single characters from text
    document = re.sub(r'\s+[a-zA-Z]\s+', ' ', document)

    # Converting to Lowercase
    document = document.lower()

    # Word tokenization
    tokens = document.split()

    # Drop words
    tokens = [word for word in tokens if len(word) > 3]

    # Lemmatization using NLTK
    lemma_txt = [stemmer.lemmatize(word) for word in tokens]

    # Remove stop words
    lemma_no_stop_txt = [word for word in lemma_txt if word not in en_stop]

    clean_txt = ' '.join(lemma_no_stop_txt)

    return clean_txt
```

0.3.3 Word Frequency Count and Intersection

We use Part-Of-Speech Tagger (POS Tagger) to identify all the nouns in the corpora.

Based on the frequency count, top 100 nouns of each corpora have been selected. Let T_{CS} and T_{SD} denote the sets of top 100 nouns in CS and a subdomain SD. Next, for each subdomain we compute $TC_{SD} = T_{CS} \cap T_{SD}$. For each subdomain SD, we estimate the ambiguity potential of those CS words which belong to the set TC_{SD}.

Function to extract all nouns from clean corpus obtained after data preprocessing :



```python
def NounExtractor(clean_corpus):
    sentences = nltk.sent_tokenize(clean_corpus)
    for sentence in sentences:
        words = nltk.word_tokenize(sentence)
        words = [word for word in words if word not in en_stop]
        tagged = nltk.pos_tag(words)
        ans = []
        for (word, tag) in tagged:
            if tag == 'NN': # If the word is a proper noun
                ans.append(word)
    return ans
```

0.4 Results

This section represents the results of the detailed experiments performed using Python 3.9 along with some supported packages and libraries. These experiments have been executed on Windows 11 machine with Intel Core i5-8265 CPU, 8 GB RAM and a processor frequency of 1.80GHz. *FastText* model is implemented using *gensim* library. For NLP related tasks, the *nltk* package has been used. The parameters of the *fastText* model have been adjusted according to the size of the class. To extract text from the pdf files, *PyPDF2* library has been used. The similarity threshold to consider a word as ambiguous is 0.3 (approximately). This value has been selected based on the the experiments on the corpora.

0.4.1 CS *versus* CHE domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : [**product**]

A1. The algorithm initializes the qubits in a **product** state and evolves them according to a Trotterized version ofthe Hamiltonian [69].

A2. Highly volatile elements such as cadmium, mercury, selenium and thallium tend to leave the cement process either in the cement kiln dust (CKD) or in the emissions and pose less of a concern for operation of the plant or for the final cement **product.**

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (CHE)
product	0.07077429	produce, exactly, probe, next, google	production, produce, layer, rich, water

0.4.2 CS *versus* MLE domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : [**operating, control**]

A1. In theory this demonstrates a potential advantage quantum computers could have over classical computers when **operating** on a specific class of problem.

A2. More crucially, no modern weapon or equipment can be operated to its full potential, including the scope for precision targeting, unless the fundamental data banks in digital form are integrated into the weapon or equipment **operating** systems.

B1. The second method is an improvement on method 1 that utilizes mid-circuit measurements to reduce the number of **control** qubits.

B2. True modernisation would, therefore, also imply balancing up the support and logistic capabilities – fire support, engineering capability, Command, **Control**, Communication, Computers, Information, Intelligence (C4I2), transport units, base facilities, and so on – within the overall structure of the modernised army.

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (MLE)
operating	0.20992355	operation, creating, operator, native, open	operation, operational, interoperability, imperative, exploiting
control	0.08235054	controlled, contrast, multiply, decomposition, equivalent	word, command, configuration, artillery, surveillance

0.4.3 CS *versus* MAR domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : [**system, cluster**]

A1. IonQ systems support most leading quan-tum programming environments, and we executed our bench-marks on the IonQ **system** using the Qiskit provider module supplied by IonQ.

A2. The data in csv, ris and bibtex formats has been extracted on 06th August 2020 from SCOPUS for China and India separately through following search criteria TITLE (”Marine Engineering” OR ”Marine Technology” OR ”Marine Environment” OR ”Marine Auxiliary Machinery” OR ”Marine Communicate” OR ”Marine Diesel Engines” OR ”Marine Ballast Systems” OR ”Marine Engine Room Simulator” OR ”Marine Gas Turbine” OR ”Marine Fuel Injection **System**” ....)

B1. Many choices exist, and in this benchmark we choose to use the unitary Coupled **Cluster** with singles and doubles ansatz (unitary-CCSD).

B2. Maritime surveillance research and marine engineering research **clusters** were identified as developing **clusters** that expanded and received increased interest and identified areas of current research interests which allowed the quantification and visualization of changes in the entire body of shipping literature over a short period (Fiskin and Cerit, 2020).

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (MAR)
system	0.2828826	trap, technique, ionq, described, evaluate	engine, world, engineering, marine, ocenography
cluster	0.17867115	operator, orbitals, laboratory, unitary, wave electronic	measure, researcher, multiple, literature, need

0.4.4 CS *versus* PTE domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : [**operator**]

A1. The quantum algorithm for order finding is actually a vari-ation of quantum phase estimation where the chosen unitary **operator** performs modular exponentiation.

A2. With the recent developments in Data Science and Engineering Analytics, there is a greater need for petroleum engineers with an understanding of physics to take advantage of these improvements and optimize the processes we use (Anadarko's SPE 187222 Creating Value by Implementing an Integrated Production Surveillance and Optimization System – An **Operator**'s Perspective and Chevron's SPE 181437 Application of Machine Learning in Transient Surveillance in a Deep-Water Oil Field, are good examples).

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (PTE)
operator	0.043061867	cluster, unitary, operation electronic, polynomial	operation, field, performance, algorithm

0.4.5 CS *versus* ME domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : [**pipeline, machine**]

A1. In addition to estimating the fidelity of results generated by quantum execution, the suite is designed to benchmark certain aspects of the execution **pipeline** in order to provide end-users with a practical measure of both the quality of and the time to solution.

A2. [21] P. Gao, H. Qu, Y. Zhang, T. Yu, and J. Zhai, "Experimental and Numerical Vibration Analysis of Hydraulic **Pipeline** System under Multiexcitations," Shock Vib., Vol.

B1. Component-level performance metrics provide a lot of infor-mation about a **machine**, but they have at least two important limitations.

B2. One type of **machine**ry referred to here is the pump system as a single pump or groups pump.

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (ME)
pipeline	0.15745302	determine, subroutine, routine, period, machine	pipe, online, approach, pulsation, must
machine	0.07061599	service, limited, processing, experimental, technology, google	type, industrial, monitoring, especially, inlet

0.4.6 CS *versus* BME domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : [**report, environment**]

A1. We therefore use the circuits' average depth to define that ensemble's volumetric position, and we **report** average performance over the circuit ensemble.

A2. The 1990 World Health Organization (WHO) **report** on the Global Burden of Disease ranked Tuberculosis (TB) as the seventh most morbidity causing disease in the world, and expected it to continue in the same position up to 2020 [1].

B1. Hardware Programming **Environment** Google QCS Cirq Alpine Quantum Qiskit Technologies Cirq TABLE II.

B2. The study focuses on generation of the social risk spatial maps for tuberculosis incidences using socio-economic, environmental factors and incidences of Tuberculosis in GIS **environment**.

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (BME)
report	-0.10593712	recent, represent, blog, ensemble, support	control, world, national, organization, burden
environment	0.13594194	programming, google, website, cirq, microsoft	environmental, map, borne, generated, correlated

0.4.7 CS *versus* CE domain

Example sentences from the corpora showing the variation of meaning of common words of both the corpora are given below : **[strength, system]**

A1. Application-Oriented Performance Benchmarks for Quantum Computing January 3, 2022 24 where J and w are the **strength** of the interactions and the disordered fields respectively, hx; i and hz;i give the **strength** of the X and Z disordered fields at site i, n is the total number of qubits.

A2. Shannag and Moura d (2012) developed high **strength** mortar matrices contain various combinations of silica fume and fly ash, and provide a good balance between workability and **strength**.

B1. The **system** has an average single-qubit gate fidelity of 99.99% and average two-qubit gate fidelity of 99.70%, as measured using standard ran-domized benchmarking [38].

B2. The first type is reinforcement concrete pipe and the anther types are ferro cement pipes with different in the type of reinforcement **system**.

SIMILARITY SCORES AND MOST SIMILAR WORDS (in respective domains)

Words	Similarity Score	Most Similar Words (CS)	Most Similar Words (CE)
strength	0.1668852	string, structure, distribution, output	compressive, high, stress, tensile, mortar
system	0.08076814	trapped, technique, ionq, described, evaluate	reinforcement, cost, form, wire, diameter

0.4.8 Limitations of the Approach

The datasets used are mostly the research papers dealing with more than one domain, hence the results extracted may not be exact. The value of the parameters passed

to *fastText* model for getting the above results are :
embedding_size : 300, window_size : 5, min_word : 4, down_sampling : 1e-2, workers : 4, sg : 1, epochs : 100.
The above values are considered after observing the results of multiple experiments. However, the model could be further improved to get more accurate results. Also, the sentences are extracted from the corpus directly which contains the given word in it (words used in same context). But there is a possibility that the words in a file can be used in different contexts.

0.5 Conclusion

This project demonstrated the applicability of *fastText* algorithm for detection of domain specific ambiguity in natural language requirements. A list of frequently used common words were extracted and the ambiguity potential was estimated based on multiple experiments. For any two ambiguous words, most similar words and some example sentences were presented to highlight its domain specific interpretation. The model is then deployed using *flask*.

Link to Github repository