Identification of Intra-Domain Ambiguity using Transformer-based Machine Learning

Ambarish Moharil
Eindhoven University of Technology
Eindhoven, North Brabant, The Netherlands
a.moharil@student.tue.nl

Arpit Sharma
Indian Institute of Science Education and Research
Bhopal, Madhya Pradesh, India
arpit.sharma@iiserb.ac.in

ABSTRACT

Recently, the application of neural word embeddings for detecting cross-domain ambiguities in software requirements has gained a significant attention from the requirements engineering (RE) community. Several approaches have been proposed in the literature for estimating the variation of meaning of commonly used terms in different domains. A major limitation of these techniques is that they are unable to identify and detect the terms that have been used in different contexts within the same application domain, i.e. intra-domain ambiguities or in a requirements document of an interdisciplinary project. We propose an approach based on the idea of bidirectional encoder representations from Transformers (BERT) and clustering for identifying such ambiguities. For every context in which a term has been used in the document, our approach returns a list of its most similar words and also provides some example sentences from the corpus highlighting its context-specific interpretation. We apply our approach to a computer science (CS) specific corpora and a multi-domain corpora which consists of textual data from eight different application domains. Our experimental results show that this approach is very effective in identifying and detecting intra-domain ambiguities.

KEYWORDS

Requirements, natural language, ambiguity, word embedding, machine learning.

ACM Reference Format:
Ambarish Moharil and Arpit Sharma. 2018. Identification of Intra-Domain Ambiguity using Transformer-based Machine Learning. In *Proceedings of The 44th International Conference on Software Engineering (ICSE 2022).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Requirements are descriptions of how the system should behave, or of a system property or attribute. The process of finding out, analyzing, documenting and checking the software requirements is called requirements engineering (RE) [24, 35]. In order to communicate the software requirements to all the project stakeholders, a software requirements document is used. The quality of requirements document determines the quality of all the other artifacts that are created during the process of software development, e.g. design document, code and test cases. According to past research [7], roughly 60% of all errors in software development projects initiate during the requirements engineering phase. One of the major causes of poor quality requirements document is the ambiguity present in the natural language (NL) text [4, 19, 27]. Due to ambiguity, the technical terms used in the project may be interpreted very differently by the project stakeholders. For example, the word *virus* means a computer program which is capable of corrupting the system or destroying data when used in the context of computer science (CS). On the other hand, for a biomedical engineer, it is a microorganism that cannot grow or reproduce apart from a living cell. Similarly, for a computer scientist, the word *cookie* means a small amount of data stored on the user's machine by the web browser. In contrast, for a food engineer, cookie is a sweet biscuit. In both these examples, ambiguity may occur due to the different technical backgrounds and domain expertise of the stakeholders involved in the RE process. Recently, the use of neural word embeddings based techniques for identifying cross-domain ambiguities in NL textual requirements have gained a significant attention. In [12], authors have used the Word2vec model to estimate the ambiguity potential of typical CS words when they are used in different domains. This approach was further extended to estimate the variation of meaning of dominant shared terms in different domains and rank them by ambiguity score by comparing the list of most similar words in each domain-specific model [13, 14]. In [31], authors have measured the ambiguity potential of most frequently used CS words when they are used in other application areas or subdomains of engineering, e.g. aerospace, civil, petroleum, biomedical, and environmental etc. An approach based on linear transformation of embedding spaces has been proposed in [25] for producing a ranked list of potentially ambiguous terms for a given set of domains. All these approaches use Wikipedia crawling for creating the domain-specific NL text corpus for every application domain.

Although these approaches can be very effective in identifying ambiguous terms between documents of different domains (cross-domain ambiguity), they fail to identify and detect the terms that have been used in different contexts within the same application domain, i.e. intra-domain ambiguity. For example, in case of biomedical domain, the word *cell* may be used to denote a device used for producing electrical energy from chemical energy as well as for representing the smallest basic unit of a plant or animal. In such a scenario, the above mentioned approaches would not be able to detect and highlight this difference. Additionally, existing approaches also fail to detect terms that have been used in different

ICSE 2022, May 21–29, 2022, Pittsburgh, PA, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

contexts in a requirements document of an interdisciplinary project. For instance, the word *Plant* may be used in the first section or paragraph of the document to represent a living thing that grows in the earth and has a stem, leaves, and roots, i.e. in the context of environmental engineering domain. On the other hand, the second paragraph or section may use it to denote a factory or a place where power is produced, i.e. in the context of petroleum engineering domain. Existing approaches fail to identify that the same term may appear in two or more different semantic roles in a requirements document. Finally, presence of intra-domain ambiguities can also create frustration and distrust during the requirements elicitation meetings [13].

This paper proposes an approach for identifying terms that have been used in different contexts within the same application domain or in a requirements document of an interdisciplinary project. Our technique is based on the idea of bidirectional encoder representations from Transformers (BERT) [10]. BERT generates a language model by reading the entire sequence of words at once. This allows the model to have a deeper sense of language context and flow. It represents individual words as real-valued vectors in a predefined vector space where vector values are learned based on the usage of words. It allows computing the semantic relationship between words in a vector space by applying the *cosine* similarity.

We create a real-valued vector for every instance of a candidate term present in the document using the BERT embeddings model. Next, we use K-means clustering which involves grouping together of instances of a term which have been used in a similar context. Roughly speaking, if the clustering algorithm returns K clusters for a term t, this represents that the term t has been used in K different contexts in the requirements document. For each cluster of a term t, we obtain a list of its most similar words and also provide some example sentences from the corpus highlighting its context-specific interpretation. We have applied our technique to a CS corpus created by crawling the CS category on Wikipedia. The results of our experiments show that our technique can be very effective in identifying intra-domain ambiguities in NL text corpus. More specifically, our technique is able to detect and identify terms that have been used in different contexts in the CS application domain. Additionally, we have also applied our approach to a hypothetical project involving multiple stakeholders from different application domains, i.e. a multidisciplinary project or a multi-domain corpora. The corpus for this project has been obtained by crawling the individual domain categories on Wikipedia and merging all the text data to create a single text file. Our findings demonstrate that even for a project involving multiple domains, this technique can be very helpful for identifying terms that have been used in different contexts in the same document.

2 RELATED WORK

2.1 Ambiguity in RE

Several works have been proposed in the literature to detect ambiguous words, terms and sentences in NL text [3, 4, 20, 21, 37]. These papers focus on domain independent ambiguous words or constructions in NL text. In [15, 16], graph-based approaches have been proposed to detect pragmatic ambiguities in NL requirements. The main idea is to use a graph-based modeling of the background knowledge of different readers, followed by a shortest-path search algorithm to model the pragmatic interpretation of a requirement. A detailed comparison of different pragmatic interpretations confirms if a requirement is ambiguous or not. In [2], an automated and tool-supported approach for checking conformance to requirements templates has been proposed for reducing ambiguity in NL requirements. This approach has been applied to four case studies from different domains, using two different templates. In [8, 29], authors translate requirements into formal models to resolve the issue of ambiguity. The role of contextual and situational ambiguity in requirements elicitation interviews has been extensively studied in [17, 18, 36]. In [30], indices of ambiguity have been defined and calculated using the knowledge base of natural language processing (NLP) system. An approach based on NLP and information visualization techniques has been proposed in [9] for pinpointing terminological ambiguities between viewpoints as well as missing requirements. NL understanding methodology has also been used for ambiguity detection in [26, 28]. Identification of potentially nocuous ambiguities using machine learning (ML) based techniques has been investigated in [38, 39].

2.2 Word Embeddings for RE

In [12], a technique based on word embeddings and Wikipedia crawling has been proposed to detect domain-specific ambiguities in natural language text. Authors have used the Word2vec model to estimate the ambiguity potential of typical computer science words when they are used in different domains. 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 [13, 14]. The authors have also proposed an algorithm to rank the terms by ambiguity score. This method has been evaluated on seven potential elicitation scenarios involving five domains. In [31], authors have measured the ambiguity potential of most frequently used computer science (CS) words when they are used in other application areas or subdomains of engineering. In [25], authors create a unified embedding space which helps in the identification of potentially ambiguous words. Word embeddings have also been used for automatic glossary term extraction [32, 34], clustering of glossary terms [5], topic modeling [22], and requirements classification [6, 23]. A novel language resource to aid in finding semantic relations between requirements has been proposed in [1]. Finally, word embeddings models for software engineering domain have been proposed in [11, 33].

3 PRELIMINARIES

This section introduces some preliminaries that are needed for the understanding of the rest of this paper.

3.1 Bidirectional Encoder Representations from Transformers

BERT is a language representation model which is bidirectionally trained. This allows BERT to have a deeper sense of language context compared to the single-direction language models. BERT uses a novel technique called Masked Language Model (MLM) which randomly masks words in the sentence and then it tries to predict them. Masking involves looking in both directions thus using the full context of the sentence for predicting the masked word. Unlike LSTMs, BERT is based on the Transformer model architecture which applies an attention mechanism to understand relationships between all words in a sentence.

4 APPROACH

In this section we discuss our approach to evaluate intra-domain contextual ambiguity. We use this approach over a domain-specific CS corpus (C_{CS}) and a multi-domain corpus (C_{MD}). In the first step we perform *Wikipedia Crawling* followed by *Data Pre-processing* in the second step. The third step involves extracting the *Contextual Embeddings* of the selected target term. In the final step we cluster all the sentences consisting of the target term into K distinct clusters. After obtaining these K clusters, we predict the context words associated with the sentences aggregated into these K clusters.

4.1 Wikipedia Crawling

Wikipedia crawling provides us with two corpora, the domain-specific CS corpus (C_{CS}) and the multi-domain corpus (C_{MD}). C_{MD} is a well curated collection of the following domains: Computer Science (C_{CS}), Ceramic Engineering (C_{CE}), Petroleum Engineering (C_{PE}), Biomedical Engineering,(C_{BE}) Industrial Engineering (C_{IE}), Marine Engineering (C_{ME}), Military Engineering (C_{MIE}) and Environmental Engineering (C_{EE}). Each of the above-mentioned domains selected for scraping are structured as a tree [14]. The nodes of this tree are the subcategories and the leaves are the number of pages. To get a visual representation of the same, the reader can check the following link[1]. To crawl the subcategories indexed in the Wikipedia tree, we implement an algorithm which given the root node performs the breadth first search to parse the subcategories until all the leaves of each and every node are retrieved. The maximum number of pages that can be downloaded for a category has been restricted to $10,000$. This restriction is helpful in retrieving the domain-specific data. All the subcategories for every single root node were verified using PetScan[2].

4.2 Data Pre-Processing

In this step, we perform pre-processing on the corpus by: a) *Removing the stopwords.* All the stopwords[3] are filtered out using the NLTK toolkit[4]. It is to be noted that certain pre-defined stopwords like ".", "++" are retained due to their semantic importance and their impact on the general interpretation of a particular word. For example, "++" is important because of its co-occurrence in terms like "$C++$" etc. b) *Sentence Tokenization.* This is a very essential step for obtaining contextual vectors of the words in our corpus. We identify sentences using the "sent_tokeinzer" library present in the NLTK package[5]. These sentences are then stacked sequentially in a list. We call this new corpus C'_{CS} for the Computer Science Domain and C'_{MD} for the multi domain corpus.

[1]https://en.wikipedia.org/wiki/Category:Computer_science
[2]https://petscan.wmflabs.org/
[3]https://gist.github.com/sebleier/554280
[4]https://www.nltk.org/
[5]https://www.nltk.org/api/nltk.tokenize.htmlmodule-nltk.tokenize

4.3 Extracting Contextual Embeddings

In order to obtain the contextual vector embeddings of a target term t, we feed all the sentences consisting of the word t to our BERT model. The "[CLS]" and "[SEP]" token is appended at the start and the end of every sentence, respectively. The target word t is extracted from each and every sentence and the corresponding vector embedding of the target word t is collected from the BERT Embedding algorithm and stored in a "*vector_mat_list*" of dimension $n \times 768$. Each of the first occurrence of the target word t in the sentence list of dimension $n \times 1$ is then labeled and stored in a "*label_list*" of dimension $n \times 1$. Correspondingly, the vectors of these n occurrences of the target word t are then stored in a "*vector_mat_list*" of dimension $n \times 768$ (as the dimension of a single BERT vector is 1×768). Following this, we create a target similarity matrix T of dimension $n \times n$. To do this, we first calculate the cosine similarity score of each target vector t_i in the *vector_mat_list* of dimension $n \times 768$ with every other target vector. Let us assume that we have two word embedding vectors ω' and ω'', then the cosine angle between these two word embedding vectors is calculated using Equation (1).

$$\cos(\overrightarrow{\omega'}, \overrightarrow{\omega''}) = \frac{\overrightarrow{\omega'} \cdot \overrightarrow{\omega''}}{|\overrightarrow{\omega'}||\overrightarrow{\omega''}|} \tag{1}$$

The range of similarity score is between 0 to 1. If the score is closer to 1 this means that the words are semantically more similar and used in almost the same context. On the other hand, if the score is closer to 0 it means that the words are less related to each other. Using these similarity scores we form a target similarity matrix T of dimension $n \times n$ where the cosine angle between t_i and t_i is always 1.

Table 1: Number of Pages and Words

Domain Name	Pages	Words
Computer Science	5601	5780693
Ceramic Engineering	388	359350
Petroleum Engineering	484	455619
Biomedical Engineering	1115	1547064
Industrial Engineering	1382	1292052
Marine Engineering	2029	1188051
Military Engineering	1540	1525956
Environmental Engineering	4601	5018976

4.4 Clustering the Contextual Vector Embeddings

Once the target similarity matrix T is obtained, it is fed to the *K-means* clustering algorithm. We calculate the WCSS (within cluster similarity score) for K in the range of 1 to 11. The algorithm automatically considers the minimum value which lies at the elbow point as the optimal K value for the respective context word. We then divide the sentences in the *sentence_list*, the labels in the *label_list* and the corresponding vectors in the *vector_mat_list* into these K clusters, predicted by the *K-means* algorithm. Let C_i represent the i^{th} cluster of the K clusters. In order to understand the K contextual semantic definitions of a target term t, we predict the context words in the clusters $C_1, C_2, ..., C_K$. Let us denote the target word t occurring in cluster C_i as t_{C_i}. Suppose there are m sentences in cluster C_1, then we predict the context words for each of these m occurrences of the term t_{C_1}. We will give a brief overview of the context words predicted within each of the K clusters which were predicted for a target term t in section V. Each of the m occurrences of the word t_{C_1} are appended in a file along with the corresponding sentence in which it has been used and the predicted context words. For cosine similarity, we keep a threshold of 0.45 which has been obtained from our detailed experimental evaluation. All the words with a similarity score greater than 0.45 are documented in the respective text file. Finally, for each target term t, we obtain K files as an output. The total number of files depends upon the number of contexts. In other words, the total number of files produced by the algorithm are equivalent to the total number of contexts in which the term t has been used or simply the total number of clusters predicted by the *K-means* algorithm. As mentioned earlier, each of the K files consists of the target word t, the corresponding sentences in which the target term t has been used and the predicted context words.

5 EXPERIMENTAL EVALUATION

This section discusses the results and the outcomes of the experimentation performed over the two corpora, i.e. C'_{CS} and C'_{MD}. The detailed report of our experiments including the replication package can be found in this repository[6]

5.1 Crawled Documents

Table 1 shows the number of words and pages present in each of the corpora mined from the Wikipedia. From Table 1, the difference in the size of the corpus for the above mentioned domains is quite evident. If we focus on the number of words, we can clearly see that there is again a considerable amount of variation between each of these domains. From this we can clearly draw a conclusion that our data is skewed to a certain extent and domains like C_{CS}, C_{ME} and C_{EE} have a dominating nature. This understanding of the dataset is extremely essential as it provides us with the extent of vocabulary the model has to deal with and also helps us in analyzing whether it can identify the most peculiar of the contexts represented by a term with a lower frequency of occurrence. Secondly, this skewed nature of our dataset poses a new challenge. The challenge is that if we consider a term which has a high frequency of occurrence in all the other domains except one, then it becomes interesting to check whether this term is used in a differing context as compared to the other domains. Additionally, does our algorithm extract that differing nature of the context represented by this term. For example, it might happen that the word "*State*" has a high frequency of occurrence in all the other domains except C_{MIE}. The word "*State*" might be majoritarily used in the following two contexts a) "*A condition*" and b) "*To express*". Now it will be interesting to see if the same word has been used in a third context, say a "*A geographical land with defined boundaries*" in C_{MIE} and does our algorithm extract this contextual representation of this word.

[6]https://github.com/ambarish-moharil/intradomain

Table 2: Implied Meaning of Words in Respective Context Clusters.

	Context Clusters	
Word	Context 1	Context 2
Function	*variable, curve, vectors, program, mathematical, loop, call, method, class, polynomial, equations*	*functioning, interaction, methodology, heuristic, organization, operation, behaviour, interaction, occurrence*
Sentences	This function sums three terms: $6x^4, -2x^3, 5$ Of these three terms, one has the highest growth rate and one the largest exponent function x,namely $6x^4$.	Stable synapses forget less easily, also it is harder to consolidate. One recent computational hypothesis involves cascades plasticity which allows the synapses to function as multiple time scales.
Language	*interpreter, syntax, compiler, compile, tool, function, function, code, program, structure, index, indentation, symbolic, logic*	*sentence, grammar, linguistic, sentences, dictionary, context, learned, arguments, grammatical, medium word, words, abstract, clause*
Sentences	C lexically is a scoped language, global scope is known as an external linkage.	To form an idea of the historical place of Jabir's alchemy and to tackle the problem of its sources, it is advisable to compare it with what remains to us of the alchemical literature in the Greek language.
Stack	*tower, pile tip, chamber, heights, racks, closet, settling, stacking, head, hub, layers, funnel, columns, blocks, piles*	*array, stacked, loop, page, lists, heap, queue, switch, frame, list, buffer, cluster, data, structure, pop, push, pull, variable, function*
Sentences	When the silo is filled or the stack is built, a layer of straw or some other dry porous substance may be spread over the surface. In the silo the pressure of the material, when chaffed, excludes air from all but the top layer; in the case of the stack extra pressure is applied by weights in order to prevent excessive heating	An abstract stack, which is a last-in-first-out structure, could be defined by three operations: push, that inserts a data item onto the stack; pop, that removes a data item from it; and peek or top, that accesses a data item on top of the stack without removal.
Application	*software, app, program, interface, users, deployment, rest, bot, tool, website*	*applying, applied, implementations, ability, experimentation, optimization, purpose, acceptance, traction, apply, applicable*
Sentences	A device using web application such as Office Web Apps.	Recently, application of neuroscience research results have also given rise to applied disciplines like neuroeconomics, neuroeducation,neuroethics,neurolaw.

5.2 CS Domain Analysis

To obtain the results for the C'_{CS} domain we have used the BERT-base-uncased model. The BERT-base-uncased model consists of 12 encoder layers, 12 attention heads, 768-hidden and 110 million parameters[7]. This BERT-base-uncased model is a pre-trained model which has been trained on a lower-cased English corpora[8]. This same architecture has been used for our experiments over the C'_{MD} corpus. In order to test the effectiveness of our approach, we have analyzed the top 150 most frequently occurring nouns in the C'_{CS} and C'_{MD}. Additionally, for C'_{MD}, we have also selected some words manually which are known to have multiple meanings in a multi-domain corpora, e.g. *concrete*. Table 2 shows some example words

[7]https://github.com/google-research/bert
[8]https://yknzhu.wixsite.com/mbweb

Table 3: Implied Meaning of Words in Respective Context Clusters.

	Context Clusters		
Word	Context 1	Context 2	Context 3
Cell	*organism, life, root, membrane, chromosomes, cellular, bodies, blood, radiation, molecular, nucleas, tissues*	*portable, cellular, phone, battery, network, tower, radio, tablet, information, wireless, smart, device*	*results, list, segment, table, matrix, storage, placed, junction, vectors, values center, entity, unit*
Sentences	Advances in microscopy also profoundly impact biological thinking. In early 19th century, large number of biologists pointed it to the central importance of a cell.	Additionally, connected cars may use WiFi, Bluetooth and communicate with the onboard consumer devices and cell phone networks.	Another very simple implementation technique, usable when the keys are restricted to a narrow range, is direct addressing into an array: the value for a given key k is stored at the array cell
Object	*class, contruct, java, oriented, method, static, variable*	*method, class, item, body, classes, instance*	*item, container, artifact, device*
Sentences	A method in object-oriented programming (OOP) is a procedure associated with a message and an object. Data is represented as properties of the object, and behaviors are represented as methods.	For example, an object in the Employee class might contain (either directly or through a pointer) an object in the Address class.	In humans, we have accomplished motor responses. Spatial planning movement, speech production, complex motor movements aspects. Consciousness awareness is whether something external object is something that is present within oneself.

from the CS domain which have been used in two different contexts. For every context in which a target word has been used, we also present its most similar words and an example sentence from the corpora highlighting the context-specific meaning of the target word. Similarly, Table 3 presents some example words from the CS domain which have been used in three different contexts.

Let us examine the word *Cell* listed in Table 3. Here, it can be observed that the word *Cell* has been used in three different contexts. From all the context words predicted and aggregated by our algorithm, we draw a conclusion that the first context defines the use of the word *Cell* as a *biological cell.* The corresponding sentences clustered with this contextual definition provide a suitable empiric for us to believe that the model has clearly discerned this definition from all the other occurrences of the same word. Secondly, we see that the word *Cell* is also used to denote a *cell phone* or a *mobile phone.* All the aggregated context words point us in this direction. Keeping our discussion confined to the C'_{CS} corpus, this definition of the word is quite expected. Surprisingly, the model also underscores or identifies another contextual use of the word *Cell.* The third cluster can be said to define the word *Cell* as an element of a data representation form, i.e. *a tabular cell.* This claim can be substantiated by the context words and the example sentence present in the table. For the sake of argument, one might put forward the question: why stop at three?, is it possible that there are additional contextual implications of this word?. It is important to note that unlike a large number of machine learning techniques, this problem statement has no ground truth. The entire approach is an unsupervised form of learning. The algorithm stops at three because the clustering converges at the corresponding K value when the target similarity matrix is given as an input. In order to check if there are other contextual implications of this word, one has to perform a manual analysis on the output files generated by the algorithm. This is due to the fact that there is no ground truth data and therefore an automated technique to verify these contextual clusters seems far-fetched considering the scope of this approach. For our set of experiments, we scrupulously analyzed each and every cluster, representing a respective contextual definition. Based on our analysis, we are in a position to confidently claim that the word *Cell* is used in three different contexts. Let us now look at the word *Function* shown in Table 2. Our approach predicts that it has been used in two different contexts. The first implication of the word *Function* can be interpreted as a *mathematical function* solely based on the predicted context words and the collected contextual sentences. Additionally, the word *Function* is also used in the context of *working or operating in a proper or particular way.* The example sentences help us in understanding this differentiation. Similarly, the word *Language* is also used in two different contexts. The algorithm identifies the first implication of the word as a *programming language* and then goes on to identify the second contextual implication of the word *Language* as a *medium of communication.* Another interesting word is *Stack* which has been used in two different contexts. Looking at the words predicted in context 1, we imply the definition of the word *Stack* as an *arrangement of objects over one another.* On the other

Table 4: Implied Meaning of Words in Respective Context Clusters.

	Context Clusters			
Word	Context 1	Context 2	Context 3	Context 4
Bank	*shore, bluff, ravine, downstream, faces, sides, confluence, hillside, delta, rainfall, reservoir, pond, rivers, channel*	*loft, rise, anfled, depression, base, fencing, moat, dyke, barrier, shaft, slip, strip, territory, spiral, holes, guts, walls, stones*	*banking, institution, financial, stakeholder, stocks, credit, market, sell, government, financing, firms, union, account, centers, reserves, money, monetary, payment, loans*	*power, store, portable, storage, battery, cell, blood, repository, unit, lithium, batteries, electricity*
Sentences	Location and geology. The Drake Well is located in Cherry tree Township, Venango County in northwestern Pennsylvania. situated on the flats 150 feet 46 m from the east bank of Oil Creek	Most of the land consisted of a simple ditch as an obstacle. On the plains this was also a water channel or moat, behind which was an earthen bank made of the spoil from excavating the ditch.	Marcos Barbosa Pinto was the chief of staff for Demian Fiocca, President of BNDES during the Luis Inacio Lula da Silva administration. He was invited to serve as president of the bank.	Battery bank, there have been significant technical advances in battery technology in recent years, and more are to be expected in the future.
Drive	*driving, driven, swing, push, adjusting, hit, pumped, run, cruise, impetus, track, predict, deploy, pitch, manage, react, lead, persist, push, encourage*	*motor, axle, gear, gearbox, fan, valve, reversal, thrust, feed, transmission, steamer, mechanism, shaft, bracket, springs, screw, dam, mounted, grid, assembly, propulsion*	*driving, cars, highway, travelling, road, automobile, vehicle, travel, around, transit, road, headlights, racing, pull, cruise, parking, slowly, highways, roads, traffic*	*partition, ram, discs, disk, memory, optical, solid, state, slot, cd, dvd, graphics, moving, hardware, download, os, rom, kernel, processor,card, pen, chip, port, usb*
Sentences	Sensors and instrumentation drive the central forces of innovation, not only for Industry 4.0, but also for other smart mega-trends, such as smart production, smart mobility, smart homes, smart cities and smart factories.	This form of power transmission is called a Zdrive because the rotary motion has to make two right angle turns, thus resembling the letter Z. This name is used to differentiate the arrangement of a drive to that of the Ldrive	Immediately after his implant, Naumann was able to use his imperfectly restored vision to drive slowly around the parking area of the research institute.	The practitioners of big data analytics processes generally hostiles as lower shared storage, preferring directattached storage or DAS. Various forms of solid state drives (SSD) are high capacity SATA disk which are buried inside the parallel processing nodes.

hand, context 2 confirms that it has also been used to denote a *data structure in computer science*. Finally, the word *application* forms a very strong case for concluding that our approach is effective. Our technique systematically identifies the two contexts for this word. In the first context it has been used to denote an *app* or a *software*. The second context uses it for *the action of putting something into operation*. First cluster aggregates all the sentences which use the noun form of the word and the second cluster aggregates all the sentences where it is used as a verb.

5.3 Multiple Domain Analysis

As discussed in the previous subsection, we have used the same BERT-base-uncased model to perform experiments on the C'_{MD} corpus using our approach. The C'_{MD} corpus is a mixed corpus consisting of eight different application domains. The aim of this experimentation was to verify whether our approach is able to detect the contexts of the target words other than the one's represented in the CS corpus, i.e. C'_{CS}. Table 4 and Table 5 show some example words that have been used in multiple contexts in the C'_{MD} corpus. Let us consider the word *Drive*. Our approach was able to identify two different contexts in which this word has been used in the C'_{CS} corpus: a) *to drive a vehicle*, and b) *a hard drive*. On the other hand, when tested on the C'_{MD} corpus, our algorithm was able to identify 4 clusters. The two additional contextual implications for this target word are c) *strive to make an effort* and d) *cause to function by force, like a shaft*. We have carefully examined the context words predicted in the two additional clusters (context 3 and context 4) for the C'_{MD} corpus and can conclude that our approach has quite accurately discerned these two additional contextual definitions of the word *Drive*. Another interesting example is the word *Bank* shown in Table 4. From the most similar words given in context 1 of the word *Bank*, it can be checked that the word has been used in the context of a *shore (especially of a river)*. This claim is also substantiated by the example sentence of context 1 in which it has been used. On analyzing the context 2, we see a very rare implication of the word *Bank*. Although the context words for this cluster seem very similar to that of the words aggregated in context 1, but a careful examination of the predicted context words along with the example sentence reveals that the contextual distinction made by our approach stands correct. More specifically, it has been used in context 2 to denote *a steep slope (as of a hill) or a tilted inward slope*. Two additional contexts in which the word *Bank* has been used are as follows: *a financial institution that accepts deposits and channels the money into lending activities*, and *a portable device that can store electricity for charging phones, cameras, and laptops*.

In a similar vein, it can be observed from Table 5 that the words *Concrete*, *Bug*, *Plant*, *State* and *Mobile* have been used in two different contexts. For instance, the word *Bug* has been used with two different meanings: the first meaning implying *Bug* as *an error in the programming of software or an app*, and the second one implying *Bug* as *a small insect*. Similarly, for the word *Plant*, our approach has managed to identify two different contexts: *a living thing that grows in the earth and has a stem, leaves, and roots*, and *a factory or a place where power is produced*. The approach also identifies two different meanings of the word *Concrete*. In one of the two contexts it can be clearly seen that the word *Concrete* implies *relating to or involving specific people, things, or actions rather than general ideas or qualities* and in the other context, *Concrete* implies *a hard strong building material.*

6 VALIDITY THREATS

In this section, we briefly address the most important of construct, internal, and external validity threats.

Construct Validity : The main threat to construct validity is that we have not used the standard performance measures, i.e. precision, recall, F-score etc, to assess the performance of our intra-domain ambiguity detection approach. This is primarily due to the fact that the entire approach is an unsupervised form of learning and the ground truth data is not available for this problem statement. In order to mitigate this threat, both the authors have manually gone through the results to check that the variation of meaning of commonly used terms predicted by the algorithm is correct or not. Another threat is that the corpora used in these experiments have been obtained by crawling Wikipedia pages and therefore are not representative of the requirements documents used in the industry. This threat could not be mitigated due to the unavailability of large-sized software requirements data from different application domains. For the same reason, all the previous research works [12–14, 25, 31] have also used Wikipedia crawling for creating the domain-specific NL text corpus.

Internal Validity : The main threats to internal validity is the researcher bias which may have occurred while checking if the different contexts predicted by our approach are correct or not. To mitigate this threat, both the authors have assessed the quality of the predicted contexts by analyzing the most similar words present in each cluster along with the example sentences in which they have been used. Another threat is that the algorithm could have missed some important contexts in which a term has been used by merging multiple contexts in the same cluster. As mentioned earlier, each cluster was manually assessed by both the authors to identify such issues. It is to be noted that no such issues were found in the output files.

External Validity : Since we have validated our approach on a large-sized CS corpora and a multi-domain corpora involving 8 application domains, we believe that the threats related to generalizability have been mitigated. However, we feel that in order to gain more confidence in our approach, it needs to be validated on several large-sized industrial requirements documents or user stories from different application domains.

7 CONCLUSIONS

We have proposed an approach for detecting terms that have been used in different contexts within the same application domain or requirements document. We have used BERT word embeddings model combined with K-means clustering algorithm to create and group multiple vectors for every candidate term in the corpora. We have applied our technique to a CS-specific corpora and a multi-domain corpora consisting of eight different domains.

Table 5: Implied Meaning of Words in Respective Context Clusters.

	Context Clusters	
Word	Context 1	Context 2
Concrete	*understood, tangible, practical, feasible, valuable, suitable, reinforcement, relevant, precise, logical, rigid, detailed, substantiate, explain, defining*	*sandstone, steel, stucco, pillars, substance, waters, graphite, glass, arches,tiles, cement, boulders, rock, slate, basalt, masonry, foundations, coping, brick, slab*
Sentences	Many concrete notions are reinterpreted as light. For example, the transpose matrix AT describes the transpose linear map given as A, with respect to dual bases	Newer cesspools are cast concrete which dramatically decrease any risk of collapse.All new construction in areas with-out waste sewer systems use these new precast cesspools
Bug	*error, problems, failure, mistakes, flaw, flaws, stuck, invalid, anomaly, issue, errors, loop, feature, defect, disrupt*	*worm, flea, insect, cockroach, rats, pigeons, bacterial, dee, worm, pit, ant*
Sentences	Erasure sometimes happened accidentally due to a program bug that caused the loop to overwrite the memory.	An insect hotel, also known as a bug hotel or insect house, is a man-made structure created to provide shelter for insects.
Plant	*stem, leaf, leaves, root, tree, chlorophyll, cell, biology, water, moisture, sunlight, light, green, pigment*	*product, production, machines, facility, process manufacturer, industrial, vessel, boiler, material, operator, task, environment, refinery, oil, industrial, station, field, commercial, building*
Sentences	Before the invention and discovery of the concept ”nature”, ancient Greek phusis, Pre-Socratic philosophers, used words that tend to describe the natural ”way”, how a plant grows.	Such a propulsion system has a smaller footprint than a diesel only power plant with the same maximal power output, since smaller engines can be used and the gas turbine and gearbox don’t need that much additional space.
State	*Behaviour, trajectory, content, configuration, conditions, shape, behave,current, trace, pattern, representations, uncertainty, arrangements*	*statewide, jurisdiction, provincial, legislative, province, county, domestic, nation, interstate, local, tribal,county, societal, regional*
Sentences	It may necessarily invoke a change in the state of our mind.	The College includes two schools: the Melbourne School of Petroleum and Geological Engineering and the Conoco Phillips School of Geology and Geophysics, hence Oklahoma Geological Survey is a state agency mandated by the Oklahoma Constitution.
Mobile	*smartphone. ipad, cellular, devices, sms, tablets, laptops, landline, cloud, portable, blackberry, nokia, phone, personal, satellite, network, telephone*	*flexible, portable, anchored, trapped, migration, diffuse, motion, permenant, clamped, location, stuck, suspended, soluble, dense, constrained, static, rotatory, coiled, wireless*
Sentences	In case construction plans, road work erecting building, supervising workers may view "blueprints" directly displays, rather using printed paper sheets. These displays include mobile devices, smartphones tablets.	In the case of mixed oxide MOX fuel, the xenon tends to diffuse out of the plutonium-rich areas of the fuel, and it is then trapped in the surrounding uranium dioxide. The neodymium tends to not be mobile.

REFERENCES

[1] Waad Alhoshan, Liping Zhao, and Riza Batista-Navarro. 2018. Using semantic frames to identify related textual requirements: an initial validation. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM.* 58:1–58:2.

[2] Chetan Arora, Mehrdad Sabetzadeh, Lionel C. Briand, and Frank Zimmer. 2015. Automated Checking of Conformance to Requirements Templates Using Natural Language Processing. *IEEE Transactions on Software Engineering* 41, 10 (2015), 944–968.

[3] D. M. Berry and E. Kamsties. 2005. The syntactically dangerous all and plural in specifications. *IEEE Software* 22, 1 (2005), 55–57.

[4] D. M. Berry, E. Kamsties, and M. M. Krieger. 2003. From contract drafting to software specification: Linguistic sources of ambiguity. (2003).

[5] Kushagra Bhatia, Siba Mishra, and Arpit Sharma. 2020. Clustering Glossary Terms Extracted from Large-Sized Software Requirements using FastText. In *ISEC 2020: 13th Innovations in Software Engineering Conference*. ACM, 5:1–5:11.

[6] Kushagra Bhatia and Arpit Sharma. 2021. Sector classification for crowd-based software requirements. In *SAC '21: The 36th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, Republic of Korea, March 22-26, 2021*. ACM, 1312–1320. https://doi.org/10.1145/3412841.3442005

[7] Barry W. Boehm. 1981. *Software Engineering Economics*. Prentice Hall, Englewood Cliffs, NJ.

[8] Alessandro Cimatti, Marco Roveri, Angelo Susi, and Stefano Tonetta. 2011. Formalizing requirements with object models and temporal constraints. *Software and System Modeling* 10, 2 (2011), 147–160.

[9] Fabiano Dalpiaz, Ivor Van Der Schalk, and Garm Lucassen. 2018. Pinpointing Ambiguity and Incompleteness in Requirements Engineering via Information Visualization and NLP. In *Requirements Engineering: Foundation for Software Quality - 24th International Working Conference, REFSQ 2018, Utrecht, The Netherlands, March 19-22, 2018, Proceedings (LNCS 10753)*. Springer, 119–135.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*. Association for Computational Linguistics, 4171–4186.

[11] Vasiliki Efstathiou, Christos Chatzilenas, and Diomidis Spinellis. 2018. Word embeddings for the software engineering domain. In *Proceedings of the 15th International Conference on Mining Software Repositories, MSR 2018, Gothenburg, Sweden, May 28-29, 2018*, Andy Zaidman, Yasutaka Kamei, and Emily Hill (Eds.). ACM, 38–41. https://doi.org/10.1145/3196398.3196448

[12] Alessio Ferrari, Beatrice Donati, and Stefania Gnesi. 2017. Detecting Domain-Specific Ambiguities: An NLP Approach Based on Wikipedia Crawling and Word Embeddings. In *25th IEEE International Requirements Engineering Conference Workshops (REW)*. 393–399.

[13] Alessio Ferrari and Andrea Esuli. 2019. An NLP approach for cross-domain ambiguity detection in requirements engineering. *Autom. Softw. Eng.* 26, 3 (2019), 559–598.

[14] Alessio Ferrari, Andrea Esuli, and Stefania Gnesi. 2018. Identification of Cross-Domain Ambiguity with Language Models. In *5th International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*. 31–38.

[15] Alessio Ferrari and Stefania Gnesi. 2012. Using collective intelligence to detect pragmatic ambiguities. In *20th IEEE International Requirements Engineering Conference*. 191–200.

[16] Alessio Ferrari, Giuseppe Lipari, Stefania Gnesi, and Giorgio Oronzo Spagnolo. 2014. Pragmatic ambiguity detection in natural language requirements. In *1st IEEE International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*. 1–8.

[17] "Alessio Ferrari, Paola Spoletini, and Stefania Gnesi". 2015. Ambiguity as a resource to disclose tacit knowledge. In *23rd IEEE International Requirements Engineering Conference (RE)*. 26–35.

[18] Alessio Ferrari, Paola Spoletini, and Stefania Gnesi. 2016. Ambiguity and tacit knowledge in requirements elicitation interviews. *Requirements Engineering* 21 (2016), 333–355.

[19] Vincenzo Gervasi, Alessio Ferrari, Didar Zowghi, and Paola Spoletini. 2019. Ambiguity in Requirements Engineering: Towards a Unifying Framework. In *From Software Engineering to Formal Methods and Tools, and Back - Essays Dedicated to Stefania Gnesi on the Occasion of Her 65th Birthday (LNCS 11865)*. Springer, 191–210.

[20] Benedikt Gleich, Oliver Creighton, and Leonid Kof. 2010. Ambiguity Detection: Towards a Tool Explaining Ambiguity Sources. In *Requirements Engineering: Foundation for Software Quality (REFSQ)*. Springer Berlin Heidelberg, 218–232.

[21] Stefania Gnesi, Giuseppe Lami, and Gianluca Trentanni. 2005. An automatic tool for the analysis of natural language requirements. *Comput. Syst. Sci. Eng.* 20, 1 (2005).

[22] Kim Julian Gülle, Nicholas Ford, Patrick Ebel, Florian Brokhausen, and Andreas Vogelsang. 2020. Topic Modeling on User Stories using Word Mover's Distance. In *7th IEEE International Workshop on Artificial Intelligence for Requirements Engineering, AIRE@RE 2020, Zurich, Switzerland, September 1, 2020*. IEEE, 52–60. https://doi.org/10.1109/AIRE51212.2020.00015

[23] Tobias Hey, Jan Keim, Anne Koziolek, and Walter F. Tichy. 2020. NoRBERT: Transfer Learning for Requirements Classification. In *28th IEEE International Requirements Engineering Conference, RE 2020, Zurich, Switzerland, August 31 - September 4, 2020*. IEEE, 169–179.

[24] M. Elizabeth C. Hull, Ken Jackson, and Jeremy Dick. 2005. *Requirements Engineering* (second ed.). Springer.

[25] Vaibhav Jain, Ruchika Malhotra, Sanskar Jain, and Nishant Tanwar. 2020. Cross-Domain Ambiguity Detection using Linear Transformation of Word Embedding Spaces. In *Joint Proceedings of REFSQ-2020 Workshops, Doctoral Symposium, Live Studies Track, and Poster Track co-located with the 26th International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2020), Pisa, Italy, March 24, 2020 (CEUR Workshop Proceedings, Vol. 2584)*. CEUR-WS.org. http://ceur-ws.org/Vol-2584/NLP4RE-paper7.pdf

[26] Nadzeya Kiyavitskaya, Nicola Zeni, Luisa Mich, and Daniel M. Berry. 2007. Requirements for Tools for Ambiguity Identification and Measurement in Natural Language Requirements Specifications. In *Proceedings of the 10th Workshop on Requirements Engineering (WER)*. 197–206.

[27] Nadzeya Kiyavitskaya, Nicola Zeni, Luisa Mich, and Daniel M. Berry. 2008. Requirements for tools for ambiguity identification and measurement in natural language requirements specifications. *Requir. Eng.* 13, 3 (2008), 207–239.

[28] "Nadzeya Kiyavitskaya, Nicola Zeni, Luisa Mich, and Daniel M. Berry". 2008. Requirements for tools for ambiguity identification and measurement in natural language requirements specifications. *Requirements Engineering* 13, 3 (2008), 207–239.

[29] L. Kof. 2010. From Requirements Documents to System Models: A Tool for Interactive Semi-Automatic Translation. In *Proceedings of the 18th IEEE International Requirements Engineering Conference*. 391–392.

[30] Luisa Mich. 2001. On the Use of Ambiguity Measures in Requirements Analysis. In *Proceedings of the 6th International Workshop on Applications of Natural Language for Information Systems (NLDB)*. 143–152.

[31] Siba Mishra and Arpit Sharma. 2019. On the Use of Word Embeddings for Identifying Domain Specific Ambiguities in Requirements. In *27th IEEE International Requirements Engineering Conference Workshops (REW)*. 234–240.

[32] Siba Mishra and Arpit Sharma. 2020. Automatic Word Embeddings-Based Glossary Term Extraction from Large-Sized Software Requirements. In *Requirements Engineering: Foundation for Software Quality - 26th International Working Conference, REFSQ (LNCS 12045)*. Springer, 203–218.

[33] Siba Mishra and Arpit Sharma. 2021. Crawling Wikipedia Pages to Train Word Embeddings Model for Software Engineering Domain. In *ISEC 2021: 14th Innovations in Software Engineering Conference, Bhubaneswar, Odisha, India, February 25-27, 2021*. ACM, 18:1–18:5. https://doi.org/10.1145/3452383.3452401

[34] Siba Mishra and Arpit Sharma. 2021. A Generalized Semantic Filter for Glossary Term Extraction from Large-Sized Software Requirements. In *ISEC 2021: 14th Innovations in Software Engineering Conference, Bhubaneswar, Odisha, India, February 25-27, 2021*. ACM, 4:1–4:9. https://doi.org/10.1145/3452383.3452387

[35] Klaus Pohl. 2010. *Requirements Engineering - Fundamentals, Principles, and Techniques* (first ed.). Springer.

[36] Paola Spoletini, Alessio Ferrari, Muneera Bano, Didar Zowghi, and Stefania Gnesi. 2018. Interview Review: An Empirical Study on Detecting Ambiguities in Requirements Elicitation Interviews. In *Requirements Engineering: Foundation for Software Quality - 24th International Working Conference, REFSQ 2018, Utrecht, The Netherlands, March 19-22, 2018, Proceedings (LNCS 10753)*. Springer, 101–118.

[37] William M. Wilson, Linda H. Rosenberg, and Lawrence E. Hyatt. 1997. Automated Analysis of Requirement Specifications. In *Proceedings of the 19th International Conference on Software Engineering (ICSE)*. 161–171.

[38] Hui Yang, Anne N. De Roeck, Vincenzo Gervasi, Alistair Willis, and Bashar Nuseibeh. 2010. Extending Nocuous Ambiguity Analysis for Anaphora in Natural Language Requirements. In *18th IEEE International Requirements Engineering Conference*. 25–34.

[39] "Hui Yang, Anne N. De Roeck, Vincenzo Gervasi, Alistair Willis, and Bashar Nuseibeh". 2011. Analysing anaphoric ambiguity in natural language requirements. *Requirements Engineering* 16, 3 (2011), 163–189.