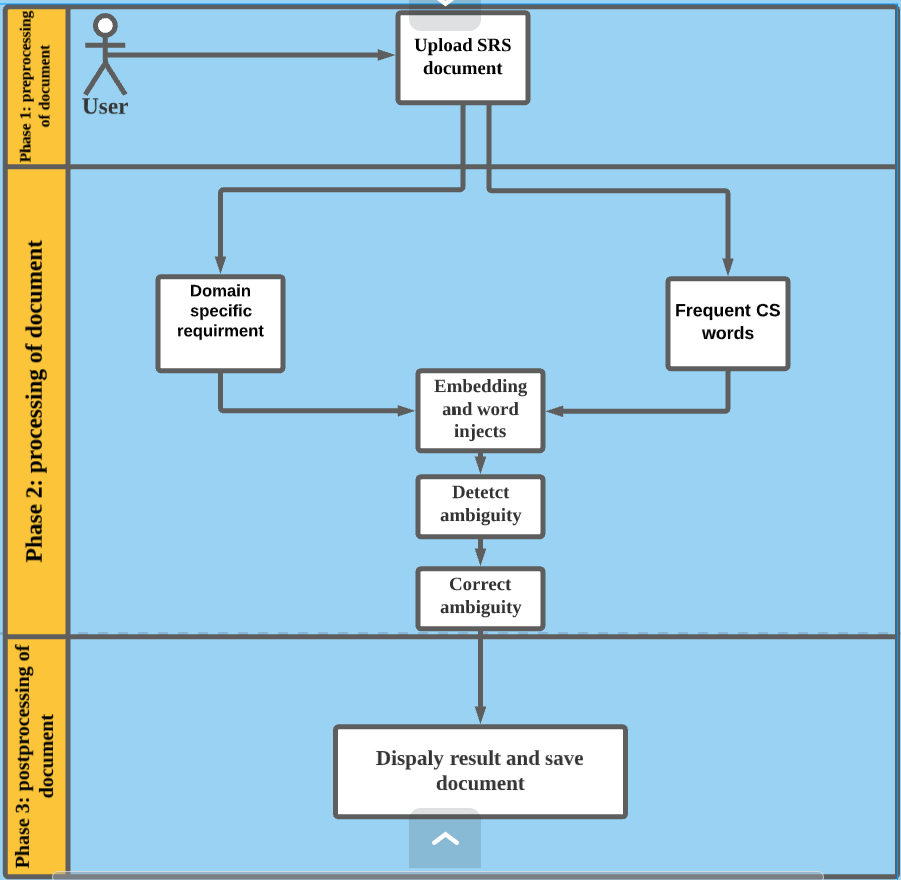
**4.1 Proposed Tool**

The aim of this research is to find a method that automatically detects and correct the ambiguous software requirements. To achieve the objectives of this research, ambiguity detection and correct approach that can automatically detect ambiguity in NL SRS document was proposed.

The proposed tool is based on one form of ambiguity in NL SRS documents. Pragmatic ambiguity. In order to detect the problems of this ambiguity. word embeddings will be used to detect ambiguity words. Then use of linked data for correcting detected ambiguity words.

The proposed approach of this research takes NL SRS document as an input, processes  
the NL SRS document then displays the result and saves the document as an output.



**Phase 1**: allows the users to upload NL software documents in (.txt) format into the system. Simply, this phase welcomes the users to define his/her requirements in English language before the processing document. This document can be ambiguous or not by comparing the similarity of embedded words. Moreover, this phase only lets user to brows NL SRS document in order to process and if the user tries to process empty document this phase prompt the user that the system don’t allow to process empty document. After uploading NL software requirements, it becomes ready to process. The main task of this phase is to make document ready to process.

**Phase 2**: once the document uploaded, this phase processes the browsed NL SRS document to detect the ambiguity in SRS. the user is suggested to access the list of frequented CS terms and choose domain specific : namely Mechanical Engineering (MCEE), Civil engineering (CIVE), Biomedical (BIEE). we focus on nouns, instead of words in general to restrict our scope.

Then the same time user processed the domain specific NL SRS it looks every word in the document to fine the similarity of CS words when used in specific domain and if the CS Words used in the document tend to have different meaning we recognized as ambiguity.

**Phase 3:** this phase presents the output of processed document. It displays the number of the detected and corrected ambiguity. The detected pragmatic ambiguity will be color as red, In this phase also allows user to save the document after detecting and correcting ambiguity words and it displays the result of detected and corrected words.

**APPROACHES USED TO DETECT AND CORRECT PRAGMATIC AMBIGUITY**

Word Embeddings =we aim to estimate the degree of ambiguity of typical computer science words (e.g., system, database, interface) when used in different domains. In this way, we investigate how the meaning of common CS words varies when these words are used in different domains. which are: Mechanical Engineering (MCEE), Civil engineering (CIVE), Biomedical (BIEE).

linked data= WordNet (API) will find all the possible interpretations or meaning of the word and will return all the interpretations related to that word to our tool and. Find a clear synonym to replace the ambiguous word.

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Word** | **Count** | **Vector** |
| 1 | machine | 2117 | 0.462 |
| 2 | design | 3437 | 0.547 |
| 3 | windows | 438 | 0.575 |
| 4 | model | 3141 | 0.420 |
| 5 | source | 3503 | 0.579 |
| 6 | application | 1355 | 0.347 |
| 7 | control | 5196 | 0.527 |
| 8 | state | 12904 | 0.573 |
| 9 | action | 2699 | 0.480 |
| 10 | cell | 2115 | 0.736 |
| 11 | function | 3330 | 0.314 |
| 12 | system | 13296 | 0.352 |
| 13 | computer | 5874 | 0.402 |
| 14 | software | 3773 | 0.374 |
| 15 | data | 4586 | 0.435 |
| 16 | time | 20412 | 0.700 |
| 17 | user | 1865 | 0.388 |
| 18 | application | 1355 | 0.347 |
| 19 | model | 3141 | 0.420 |
| 20 | information | 5844 | 0.435 |
| 21 | problem | 449 | 0.367 |
| 22 | function | 3330 | 0.314 |
| 23 | language | 9375 | 0.490 |
| 24 | algorithm | 1047 | 0.361 |
| 25 | science | 848 | 0.503 |
| 26 | university | 10195 | 0.355 |
| 27 | program | 3403 | 0.583 |
| 28 | set | 7682 | 0.405 |
| 29 | use | 14011 | 0.609 |
| 30 | method | 2397 | 0.544 |
| 31 | research | 3452 | 0.447 |
| 32 | device | 1195 | 0.612 |
|  | …….. |  |  |

Table 1 list of common used computer science words

|  |  |  |
| --- | --- | --- |
| **Words** | **Most Similar Words (CS)** | **Most Similar Words (MCEE)** |
| device | monitor, address, access, hardware | airbag, electricity, fabrication, petroleum, diagnostics |
| machine | data, semantics, pattern, class, complexity, search, analysis | construction, task, manufacturing, category, hazard, diagnostics |
| design | knowledge, network, analysis, program, processing, architecture | standardization, construction, standard, guideline, fabrication, equipment |

Table 2 Most Similar Words For CS And MCEE.

CS versus MCEE domain: The most frequently used CS words which have a different meaning in the domain of MCEE are shown in Table 2. Example sentences showing the variation of meaning of CS words machine and device are given below:

|  |
| --- |
| CS. machine: A finite state machine just looks at the input signal and the current state: it has no stack to work with. |
| MCEE. machine: An air classifier is an industrial machine which separates materials by a combination of size, shape, and density. |
| CS. device: A simple 2D mouse may be considered a navigation device. |
| MCEE. device: A thermal cutout safety device is required to prevent overheating of the heating element. |

|  |  |  |
| --- | --- | --- |
| **Words** | **Most Similar Words (CS)** | **Most Similar Words (BUE)** |
| window | drive, file, folder, linux, driver | curtain, pit, clothes, fireplace, tile |
| model | method, data, analysis, simula tion, prediction | mechanic, law, matter, thermodynamics |
| source | tool, application, specification, library | cooling, earth, noise, generator |
| device | hardware, access, monitor, address | pump, chiller, motor, conditioner |
| application | development, database, software, tool, architecture, source | apparatus, duct, leakage, refrigerant, combustion |
| control | code, source, database, interface, storage, requirement | ventilation, rating, consumption, supply, electricity |
| design | program, network, processing, architecture, knowledge, anal ysis | edition, issue, communication, access, activity |
| system | network, design, development, software, database, program | power, area, control, ventilation, consumption |

Table 3 Most similar words for CS AND BUE.

CS versus BUE domain: The most frequently used CS words which have a different meaning or interpretation in the domain of BUE are shown in Table 3. Example sentences showing the variation of meaning of CS words window and source are given below :

|  |
| --- |
| CS. window: An active window is the currently focused window in the current window manager or explorer. |
| BUE. window: means an opening in the wall people use to gain light or air and allow people to see outside |

|  |
| --- |
| CS. source: Auto Keras is an open-source python package for neural architecture search. |
| BUE. source: An absorption refrigerator is a refrigerator that uses a heat source . |

|  |  |  |
| --- | --- | --- |
| **Words** | **Most Similar Words (CS)** | **Most Similar Words (BIEE)** |
| machine | process, code, event, computation, source, analysis | stimulation, blood, movement, heart, material |
| system | network, design, software, development, database, program | neuroscience, time, neuron, activity, breast, cell, brain,rate |
| action | operation, decision, configuration, object, pattern, channel | neuron, anatomy, shape, image, flow, membrane |
| state | class, algorithm, automaton, domain, model | activity, breast, rate, neuron, image |
| cell | value, rank, error, size, input | chemical, brain, stem, neuro science, image |
| function | case, operator, parameter, array, variable, integer, input, define | anatomy, momentum, equation, condition, cell, coordinate |

Table 4 Most Similar Words for CS AND BIEE.

CS versus BIEE domain: The most frequently used CS words which have a different meaning in the domain of BIEE are shown in Table 4. Example sentences showing the variation of meaning of CS words machine, cell and function are given below:

|  |
| --- |
| CS. machine: User-accessible registers can be read or written by machine instructions. |
| BIEE. machine: One of the most successful approaches is an external device that acts similarly to a dialysis machine. |

|  |
| --- |
| CS. cell: Apple provided a biogas-powered fuel cell and built rooftop solar photovoltaic systems. |
| BIEE. cell: Clots formed by red blood cell and platelet damage can block up blood vessels and lead to very serious consequences. |

|  |
| --- |
| CS. function: The function point metric was introduced to calculate the number of user input and output transactions. |
| BIEE. function: Restoration for the volitional motor function via an artificial neural connection |

### **Steps for ambiguity detection and correction**

The proposed tool follows 7 steps in order to detect and correct the pragmatic ambiguity in NL software documents. These steps uses both word embeddings and linked data technique, in which, each step has specific role for detecting and correcting the ambiguities of NLSRS documents.

**STEP 1: Upload NL software requirements document.**

The main role of this step is extracting or attaching the document from the user in order to be ready for processing. The uploaded document can be a collection of sentences or paragraphs. In this stage the browsed document is prepared to process.

**STEP 2: Choose specific domain and Mark each word to list of CS terminology.**

This step includes lists from the CS terminologies, and SRS document for specific domain. each CS word of the uploaded document is compared with words from the list to investigate  
how the meaning of common CS words varies when these words are used in different domains. the user is suggested to access the domain for More specifically, based on the input SRS document, this allows estimating the variation of meaning the frequent CS nouns when used in the NLSRS domain documents. The process is repeated for each domain.

**STEP 3: word embeddings and word injection.**

This step retrieves the uploaded document word by word and compares each one into the predefined list of stored CS terms. We process each SRS document by removing stop-words and by lemmatizing. We remove stop-words, i.e., common words such as conjunctions, articles and pronouns, since in this context we are not interested in these words, which do not bear a domain specific meaning. Lemmatization allows the treatment of word inflections (e.g., requirements and requirement) as a single word (i.e., requirement).

**STEP 4: Detect pragmatic ambiguity.**

This step, it checks if the matched word from the CS **terminologies**, and SRS document for specific domain the meaning varies. If the “word\_match”. For domain specific SRS differ from stored CS **terminologies.** Then, all words matched the words in the listed CS terms will be counted and recognized as pragmatic ambiguity. The counted words are stored as variable “Sentence\_Counter”. Simply, the main role of this step is to compare each word of uploaded documents to a list of CS terminology and count the matched words and store a variable called “word\_match”. If there is no matched words, displays the result as zero.

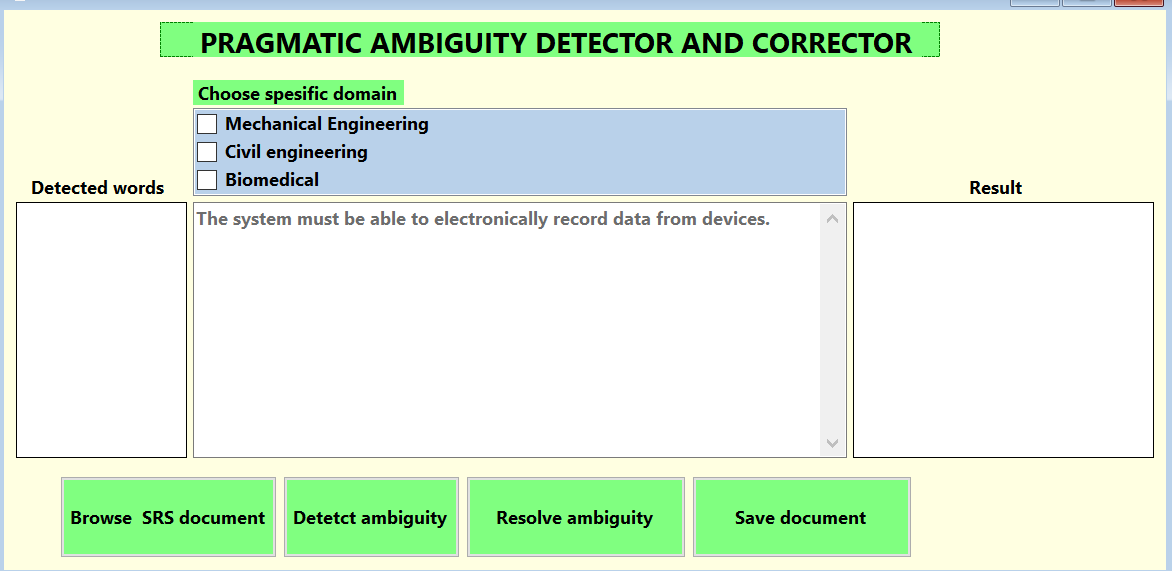
**STEP 5: Correcting pragmatic ambiguity.**

The main role of this step is disambiguationof detected ambiguity in here we implement WordNet API which will find all the possible interpretations under only Nouns of that word, when user hover the mouse in that particular word and if the number of interpretations is more than one, the WordNet will return all the interpretation related to that word to our tool and Finds a clear synonym to replace the ambiguous word.

**STEP 6: Repeat step 3, 4 and 5 until each sentences of document reaches at the end.**

**STEP 7: Display the total number of ambiguity and save the document.**

**From design**



**The system must be able to electronically record data from devices.**