Extracting Conjunction Patterns in Relation Triplets from Complex Requirement Sentence

Veera Prathap Reddy M*1, Prasad P.V.R.D*2, Manjunath Chikkamath3 and Karthikeyan Ponnalagu4
*Robert Bosch Engineering and Business Solutions Pvt Ltd, Bangalore, India
*Department of Computer Science and Engineering, K.L Education Foundation, Vaddeswaram, Andhra Pradesh, India

Abstract
Automatically extracting knowledge from complex unstructured software requirement sentences is an important research challenge. The objective would be reducing human interpretation errors that contribute to more than 50% of overall software defects. In this paper, we propose pattern based open information extraction (OIE) approach towards addressing this challenge. Our proposed approach extracts meaningful relations from natural language sentences that are considered complex with conjunctive (correlative, coordinating and subordinating) structures. Our proposed approach exploits linguistic knowledge about English language grammar to identify pattern in requirement sentence and subsequently extract information according to the grammatical function of its constituents. We propose MRAlgo, an automated multiple-relation Verb centric information extraction algorithm specifically for software requirement engineering domain that can detect every action, subject and object when linked with conjunctions. We have evaluated MRAlgo by a random sample of sentences selected from public dataset of requirement sentences having conjunctive nature and few sentences from web, and obtained high precision and recall when compared to other Open information extraction approaches.

Keywords - Multiple-relation extraction, Natural Language Processing (NLP), dependency parser, verb-based algorithm.

I. INTRODUCTION

In traditional software engineering methodologies [2][3][4][5] such as Waterfall or V-Model of development, the first phase is requirement analysis. This basically ensures capturing the project requirements in clear formal specifications. The delays and errors in terms of manual oversights caused in this phase, lead to exponential cost in the completion of the project. Often requirements are not correctly and completely stated, leaving scope of human interpretations. The eventual interpretation varies between two individuals based on their levels of experience and expertise dealing with requirements for a given domain. Software requirement specification (SRS) documents are often limited by various types of ambiguities as depicted in Fig. 1. Hence as part of analyzing the requirements, the ambiguities need to be resolved. Towards that, converting informal natural language requirements (NLRs) into formalized representation through automated means ensures consistency. The machine learning techniques can be leveraged to meet this goal. This affects the subsequent software engineering steps such as design or test specification writing for example in V-model methodology.

Software Requirement specification document contains hundreds of requirement sentences explaining functionalities or responsibilities of a system or a person. Each of such sentences may include rich
amount of conjunctions in it. A conjunction is a word that is used to connect phrases, sentences and clauses. Understanding of such complex statements by leveraging Artificial Intelligence techniques such as Natural Language Processing (NLP) is necessary. Such techniques focus on extracting formal insights from unstructured text which can be useful in automating tasks such as text categorization, text summarization, generating structured representation and answering queries [36].

An effective algorithm, which can resolve the patterns present in such complex sentences and extract every relation between subjects, objects and verbs along with their conjunctions, is required.

Natural Language Processing (NLP) is a cross disciplinary field in artificial intelligence and computational linguistics. It investigates in multiple ways to enable computers to interact with humans and understand human natural languages. Standard techniques in NLP include word segmentation, Parts of Speech (POS) tagging, word sense disambiguation (WSD) [6][7], parsing, and Named Entity Recognition (NER) [8][9].

Word segmentation enables computers in identifying and extracting valid words from a continuous stream. POS tagging helps computers to classify words into categories such as noun, verb, and adjectives. Parsing determines structure of the sentences based on POS tags. NER helps computers to recognize and classify named entities that are rigid designators [10] in texts into pre-defined categories such as proper names of persons, organizations, locations etc.

Open information extraction (OIE) is the task of generating a structured, machine-readable representation of the information in text, usually in the form of triplets. A triplet can be understood as truth-bearer, a textual expression of a potential fact (e.g., “Dante wrote the Divine Comedy”), represented in an amenable structure for computers [e.g., (“Dante”, “wrote”, “Divine Comedy”)]. The first argument is usually referred as the subject, second argument as relation and while third is considered to be the object [42]. OIE can be seen as the first step to a wide range of deeper text understanding tasks such as relation extraction, knowledge-base construction, question answering, semantic role labeling. The extracted triplets can also be directly used for end-user applications such as structured search (e.g., retrieve all triplets with “Dante” as subject).

This paper proposes MRAIgo, to extract information from complex compound sentences by identifying the pattern and extracting coherent triplets.

II. BACKGROUND

Open information extraction (OIE)[40] aims to obtain a shallow semantic representation of large amounts of natural language text in the form of verbs (or verbal phrases) and their arguments. The key goals of OIE are as follows:

- To extract the notion of entities without specially trained for a domain,
- Unsupervised extraction without necessitating any domain centric corpus for training, 3) Can easily scale to large amounts of text.

OIE was first introduced by TextRunner [43] developed at the University of Washington Turing Center headed by Oren Etzioni. Other methods introduced later such as Reverb [44], OLLIE [45], ClausIE [46] or CSD [47] helped to shape the OIE task by characterizing some of its aspects. At a high level, all of these approaches make use of a set of patterns to generate the extractions. Depending on the particular approach, these patterns are either hand-crafted or learned. Approaches such as TextRunner [43], WOEpos [51], Reverb [44], and R2A2 [49] focus on efficiency by restricting syntactic analysis to part-of-speech tagging and chunking. These fast extractors usually obtain high precision at low points of recall, but the restriction to shallow syntactic analysis limits maximum recall and/or may lead to a significant drop of precision at higher points of recall. Other approaches such as Wanderlust[42], WOEparse[51], KrakeN [48], OLLIE [45], [50] and ClausIE [46] additionally use dependency parsing. These extractors are generally more expensive than the extractors above, they trade efficiency for improved precision and recall.

In this paper, we propose MRAIgo for Verb Centric open information extraction algorithm, which makes use of dependency parsing. MRAIgo follows verb-based approach to identify the pattern based on dependency tokens and pos taggers in the sentence parsed to extract one or more coherent triplets. The reason being verb-based approach is that it will detect pattern from the grammatical functions of its constituents, which adopts easily to new patterns with no training required. Sentence processing is parallel and can process single sentences to large document collections automatically and in a scalable way.

In sentence “coding in python is good for Machine Learning”, MRAIgo extracts following triplets (coding in python, is, good), (coding in python, is good for, machine learning). Most of the patterns of [48], [50], [52] are naturally captured by MRAIgo. Moreover, MRAIgo can able to process complex compound sentences having conjunctions at subject, object and verb levels by maintaining coherency and completeness at every pattern to improve precision and recall.
We evaluated MRAIgo with random sample drawn from dataset collected from web and complex compound conjunctive sentences present in software requirement specification documents (SRS). The resulted extractions are compared with other OIE techniques named ClausIE and OpenIE[53]. We found that MRAIgo obtains significantly more coherent triplets than previous approaches at similar or higher precision and higher recall with the extracted coherent triplets.

The rest of the paper is organized as follows: Section III explains the related work. Section IV explains patterns of triplets from the different structured conjunctive sentences. Section V provides description and pseudo-code for proposed algorithm MRAIgo. Section VI provides an evaluation of the proposed algorithm. Section VII includes conclusions and future work.

III. RELATED WORK

Automatic relation extraction from unstructured texts has recently attracted considerable interest [22] [23] [26]. The main approaches used for this are described in this section as follows:

Co-occurrence approaches provide the simplest way to detect relations if the two entities are frequently collocated with each other across a collection of texts or sentences. They result in high recalls but may have poor precisions. Now they are usually compared against other methods as a baseline method [27] [28].

Link-based approaches extend co-occurrence approaches if the two entities often co-occur with a common term across a collection of corpus. They usually improve the precision but the recall rate remains low [29]. Although in theory both approaches can be applied directly to raw texts, NLP techniques are employed in virtually all cases to pre-process the text.

Machine learning approaches label and segment sentences automatically by using Hidden Markov Model [30], Conditional Random Fields [31] and Naive Bayes classifier [32]. However, they require manually annotated training data which can be expensive to obtain. In addition, they may result in a limited coverage in different domains.

Rule-based approaches use NLP techniques and templates generated manually by domain experts to identify semantic entities and extract associations connected by some specific verbs [33][34]. Standard NLP techniques such as POS tagging parsing, and NER are used to generate the dependency trees and simple co-occur relation structures, such as Entity-VerbEntity, Entity binds Entity but not Entity, are considered for relation extraction, resulting in a reasonable precision around 80% and recall around 85%. However, they are computationally costly if they are dealing with large size data [27]. In addition, most investigation of rule-based approaches has centered around specific types of relationships.

Verb-based approaches share some similarities with the rule-based approaches. They both highly rely on NLP techniques, while verb-based approaches cover a much wider range of complex relationship types [35].

OIE-based approaches such as TextRunner[43], WOE[51], Reverb[44], and R2A2[49] focus on efficiency by restricting syntactic analysis to part-of-speech tagging and chunking. These fast extractors usually obtain high precision for high confidence triplets. This is however with poor recall due to limitations of shallow syntactic analysis. Other approaches such as Wanderlust[42], WOE[51], KrakeN[48], OLLIE[45], and [50] additionally use dependency parsing making it more performance draining as a trade-off for improved precisions and recall.

IV. SENTENCE PATTERNS

The main objective of this work is to extract every relationship between entities present in the sentence. MRAIgo firstly identify the pattern of the sentence parsed and extract information from the sentence in triplet form.

A pattern is a part of a sentence that expresses some coherent piece of information, it consists of one subject, one verb and optionally of an indirect object, a direct object, a complement and one or more adverbials. Not all combinations of these constituents appear in the English language.

Sentences in requirement specifications can range from simple to complex compound sentences. MRAIgo can identify the patterns [48], [50], [52] where other algorithms can able to extract, also can extract the patterns where other existing algorithms couldn’t able to retrieve. The patterns where MRAIgo can identify, which other existing approaches can’t identify in retrieving coherent triplets are complex conjunctive structured sentences. All such sentences are evaluated, processed to improve accuracy in this paper. The following are different patterns with multiple subjects, multiple objects, and multiple predicate combinations identified as conjunction patterns. Representation of such pattern structures are mentioned in Table I

<table>
<thead>
<tr>
<th>Table I. Representation of Structures For Patterns Discussed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sent Structure</td>
</tr>
<tr>
<td>S1V1O1</td>
</tr>
<tr>
<td>SnV1O1</td>
</tr>
</tbody>
</table>
While arriving at different patterns with dependency tokens, a sentence can be constructed according to grammatical function of the constituents, we obtained 14 patterns. MRAAlgo can also extract information where governing constituent is noun, adjective as well. Each of this pattern internally includes the patterns discussed in Table I. A complete list of all patterns are listed in Table II.

Few of the spacy dependency tokens used to recognize semantic relations between words are as follows:
- **ACOMP**: adjectival complement
- **ADVCL**: adverbial clause modifier
- **ADVMOD**: adverbial modifier
- **AUX**: auxiliary
- **AUXPASS**: passive auxiliary
- **CC**: coordinating conjunction
- **CCOMP**: clausal complement
- **CONJ**: conjunct
- **CSUBJ**: clausal subject
- **CSUBJPASS**: clausal passive subject
- **DOBJ**: direct object
- **NSUBJ**: nominal subject
- **NSUBJPASS**: nominal passive subject
- **PCOMP**: complement of a preposition
- **POBJ**: object of a preposition
- **PREP**: prepositional modifier
- **XCOMP**: open clausal complement

<table>
<thead>
<tr>
<th>Table II. MRAAlgo Patterns in Sync with Other Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SIVnO1</strong> Represents sentences with single subject, having multiple conjunctions to verb and single object</td>
</tr>
<tr>
<td><strong>SIVnOn</strong> Represents sentences with single subject, single verb and object having multiple conjunctions</td>
</tr>
<tr>
<td><strong>SvVnO1</strong> Represents sentences with subject having multiple conjunctions, verb having multiple conjunctions and single object</td>
</tr>
<tr>
<td><strong>SvVnOn</strong> Represents sentences with subject having multiple conjunctions, verb having multiple conjunctions and object having multiple conjunctions</td>
</tr>
<tr>
<td><strong>SiViO1</strong> Multiple verbs which were not conjunctions, along with multiple subjects and multiple objects which were not conjunctions</td>
</tr>
</tbody>
</table>

### Table III. Conjunctions Extracted from the Sentence

<table>
<thead>
<tr>
<th>Conjunction Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VERBS</strong></td>
<td>List of all verbs from the sentence with POS as VERB</td>
</tr>
<tr>
<td><strong>VERBSCONJ</strong></td>
<td>Verbs from sentence which are linked with dependency tokens as CC (co-ordinating conjunction) and CCONJ (co-ordinating conjunctions)</td>
</tr>
<tr>
<td><strong>NOUNCONJ</strong></td>
<td>Nouns from sentence which are linked with dependency tokens as CC (co-ordinating conjunction) and CCONJ (co-ordinating conjunctions)</td>
</tr>
<tr>
<td><strong>ADJCONJ</strong></td>
<td>Adjectives from sentence which are linked with dependency tokens as CC (co-ordinating conjunction) and CCONJ (co-ordinating conjunctions)</td>
</tr>
</tbody>
</table>

V. **MRAAlgo: MULTI-RELATION EXTRACTION ALGORITHM**

The input to the proposed algorithm MRAAlgo are general sentences or sentences containing software requirement sentences. The system process each sentence to find coreferences present and resolve them using Neural-Co-ref[41]. An overview of a single iteration process is shown in Fig. 2.

MRAAlgo consists of two main tasks,
1) Data pre-processing and
2) Text processing for information extraction.

The first task carries out pre-processing functions for POS tagging and parsing. In text processing for information extraction, dependency tokens were used to identify semantic relations between words. Finally, the system generates the relations in the form of (Subject Predicate/verb Object). In the following sections, we describe each of these steps in detail.

The extracted relationships can further be visualized with Knowledge graphs to understand the entity attributes and properties. Therefore, an automatic procedure was developed and written in Python.
ADVCLVERB | Verbs from sentence which are linked with dependency tokens as ADVCL (adverbial clause modifier).
---|---
ACLVERB | Verbs from sentence which are linked with dependency tokens as ACL (finite/non-finite clause modifier).
XCOMPVERB | Verbs from sentence which are linked with dependency tokens as XCOMP (open clausal compliment).

A. Data pre-processing

Data pre-processing is to have a clean text corpus readily available for algorithm as input for extracting entities and relations that exist among entities. Every sentence is processed for co-reference resolution to replace references to the subject in the statement. The proposed algorithm determines the structure of the sentence based on POS tags and dependencies. The generated output of the algorithm consists of sentences with the corresponding POS tags and syntactic dependencies to one another.

B. Text Processing for Information Extraction

Conjunction Lists which are extracted from the sentence before we process sentences for triplet extraction were shown in Table III.

VERBSCONJ list consists of verbs which were conjunctions to each other and were connected with dependency tokens CC (Co-ordinating conjunction) and CONJ(conjunction). ACLVERB list consists of elements where verb is linked with other verbs object. The data processing procedure only focused on sentences that contain valued relationships between extracted noun phrases or chunks. Therefore, the algorithm starts with the following conditions:

- C1: Identifying verbs bearing sentence.
- C2: Listing out all dependencies that were listed to the verb either directly/indirectly to subjects.
- C3: List all dependencies that were marked as objects either directly/indirectly to the verb.

If the sentence being processed satisfies the conditions C1,C2 and C3, it is assumed to contain a valid relation. The algorithm then extracts and returns all coherent triplet SubjectVerb-Object.

The occurrence of triplet happens when a verb is having dependency with the subject entity and object entity. However, software engineers engaging in requirement engineering activity commonly use more complicated sentence structures. MRAlgo will be able to deal with simple, moderate and complex sentences having conjunctive and non-conjunctive structures. In the scope of this paper, we consider requirement sentence structures having conjunctions occurring at following scenarios:

1) Subject in the sentence having multiple conjunctions.
2) Predicates/verbs in the sentence having multiple conjunctions,
3) Objects in the sentence having multiple conjunctions.

MRAlgo will identify each and every verb with POS tagged attached to it and extract information about subjects and objects to verb with dependency tokens linked to it.
Pseudocode of MRAlgo is shown in Algorithm 1.

Line 2 of algorithm lists all tokens, which are identified as subjects to the processing verb. Line 3 consists of tokens which can be identified as objects to the verb. Line 4 consists of all verbs present in the sentence. Line 5 consists of verb conjunctions for every verb present in the sentence. Line 6 consists of noun conjunctions for every noun present in the sentence. Line 7 consists of all adjectives present in the sentence. Line 8 consists of all adverbial clause modifier verbs present in the sentence. Line 9 consists of all finite/non-finite clause modifier verbs present in a sentence. Line 10 consists of all subjects present in the sentence. Line 11 includes logic to extract subject and its corresponding conjunction list as subjects to the verb. Line 30 to 37 extracts every object that is linked to verb as direct object or by objects present in the conjunction list. Line 46 to 52 extracts triplets from the extracted subjects, verb and objects list in Subject-Verb-Object pattern. Finally algorithm lists out all triplets as list of lists as output to given sentence.

VI. EVALUATION

Table IV illustrate with examples for each pattern structure by listing out the triplets extracted from MRAlgo.

A. Extraction Effectiveness

In this section, we discuss the extraction effectiveness of MRAlgo in identifying triplets from the sentences. For example the sentence represented in the (S1V1On): Administrators should have skills in system administration, database management and deployment of Web applications explains the responsibilities of an administrator as a user of the application. This explains the valid relations between the administrator and his responsibilities. The Proposed algorithm able to extract all possible relations where conjunctions are linked with multiple subjects and objects. In another example (SnVnOn) for input sentence The data manager and administrator wants to manage jobs so as to monitor finished and upcoming jobs, schedule a new job or cancel a scheduled job. The algorithm extracted all valid triplets.

Algorithm 1 MRAlgo: Verb-Centric Triplet Extraction Algorithm

1: Input : Requirement Sentence from SRS, Output : Triplets from Sentence
3: OBJDEPS = (‘attr’, ’dative’, ’dobj’, ’oprd’)
4: VERBS = [V₁, V₂... Vₘ]
5: VERBCONJ = [[vc₁,...vcₙ],[vc₁...vcₙ]...
6: NOUNCONJ = [[Nc₁,...Ncₙ],[Nc₁...Ncₙ]...
7: ADJCONJ = [[Ac₁,...Acₙ],[Ac₁...Acₙ]...
8: ADVCLVERB = [[ADc₁,...ADcₙ],[ADc₁...ADcₙ]...
9: ACLVERB = [[AcL₁...AcLₙ],[AcL₁...AcLₙ]...
10: XCOMPVERB = [[Xc₁,...Xcₙ],[Xc₁...Xcₙ]...
11: for all eachverb Vᵢ...Vₙ in verbs do
   12: ======={Extract Subjects of the Verb}======
   13: SUBJS = [Vᵢ with SUBJDEPS ]
   14: if len(SUBJS) == 0 and Vᵢ in VERBCONJ then
   15: SUBJS = [Vᵢ with SUBJDEPS ]
   16: end if
   17: if len(SUBJS) == 0 and Vᵢ in XCOMPVERB then
   18: SUBJS = [XCOMPVERB with SUBJDEPS ]
   19: end if
   20: if len(SUBJS) == 0 AND Vᵢ in ADVCLVERB then
   21: SUBJS = [ADVCLVERB with SUBJDEPS ]
   22: end if
   23: if len(SUBJS) == 0 AND Vᵢ in ACLVERB then
   24: SUBJS = [ACLVERB with SUBJDEPS ]
   25: end if
   26: if len(SUBJS) == 0 then
   27: SUBJS = First subject of the sentence
   28: end if
   29: ======={Extract Verb Conjunctions}======
   30: if Vᵢ in VERBCONJ then
   31: Verblist = [VERBCONJ(V Cᵢ... V Cₙ)] + Vᵢ
   32: end if
   33: if Vᵢ in VerbList then
   34: for all V Cᵢ in VerbList AND Verb len([V Cᵢ with SUBJDEPS]) >0 do
   35: Remove V Cᵢ from VerbList
   36: end for
   37: end if
   38: ======{Extract Objects of Verb Vᵢ}======
   39: OBJJS = [Vᵢ with dependency tokens in OBJDEPS ]
   40: if len(OBJJS) = 0 then
   41: if Vᵢ in VERBCONJ AND V Cᵢ with OBJDEPS then
   42: OBJJS = [Objects linked to all VerbList]
   43: end if
   44: end if
   45: ={Extract Triplets from SUBJS, OBJJS, VERBS }==
   46: for all V Cᵢ in VerbList [V Cᵢ, V Cᵢ,.. V Cₙ]
   47: for all subject in SUBJS [S₁...Sₙ] do
   48: for all object in OBJJS [O₁, O₂...Oₙ] do
   49: triplet = Sᵢ, Vᵢ, Oᵢ
   50: end for
   51: end for
   52: end for
   53: end for
   54: Return Triplets

---

ISSN: 2231-2803 http://www.ijcttjournal.org Page 138
We compared MRAalgo to ClausIE and OpenIE with the random sample of 50 sentences from software requirement specifications documents (SRS)\(^1\), which includes complex compound sentences having conjunctive nature (which includes correlating, coordinating and subordinating conjunctions) and few more random sentences from web which include all patterns. Sample examples of such sentences were given in Table V.

A comparison is made between MRAalgo to single relation extraction algorithm (OpenIE) and multiple relation extraction algorithm (ClausIE). We used the absolute number of extractions, since it is infeasible to obtain the set of all correct triplets. For MRAalgo, we determined the number of nonredundant extractions, i.e., extractions not contained in other extractions. For example, MRAalgo extracts from sentence “AE remained in Princeton until his death” propositions (\textit{AE, remained, in Princeton}) and (\textit{AE, remained, in Princeton until his death}).

<table>
<thead>
<tr>
<th>Pattern No.</th>
<th>Example and Triplets Extracted</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1V1O1</td>
<td>The system displays the metadata to the data manager (system, displays metadata to, data manager) (system, displays, metadata)</td>
</tr>
<tr>
<td>SnV1O1</td>
<td>Either administrator or data manager should be good in database. (data manager, should be good in, database) (administrator, should be good in, database)</td>
</tr>
<tr>
<td>S1VnO1</td>
<td>Administrators are responsible for installing, configuring and monitoring the system. (Administrators, installing, system) (Administrators, monitoring, system)</td>
</tr>
<tr>
<td>S1V1On</td>
<td>Administrators should have skills in system administration, database management and deployment of Web applications. (administrators, should have, skills) (administrators, should have skills in, database management) (administrators, should have skills in, system administration) (administrators, should have skills in deployment) (administrators, should have skills in deployment of, web applications)</td>
</tr>
<tr>
<td>SnV1On</td>
<td>portal managers, Data managers should have basic knowledge of taxonomy and biodiversity data. (portal managers, should have basic knowledge of, taxonomy) (portal managers, should have basic knowledge of, biodiversity data) (data managers, should have basic knowledge of, taxonomy) (data managers, should have basic knowledge of, biodiversity data).</td>
</tr>
<tr>
<td>SnVnO1</td>
<td>IPT instances and GBIF portal Web services, other data source types may be configured or updated as modules. (data source types, may be configured as, modules) (portal web services, may be configured as, modules) (ipt instances, may be configured as, modules)</td>
</tr>
<tr>
<td>S1VnOn</td>
<td>The data manager wants to manage resources to import a new resource, edit a resource metadata or delete a resource. (data manager, delete, resource metadata) (data manager, wants to manage, resources) (data manager, to import, new resource) (data manager, delete, resource) (data manager, edit to manage, resources) (data manager, edit, resource metadata)</td>
</tr>
<tr>
<td>SnVnOn</td>
<td>The data manager and administrator wants to manage jobs so as to monitor finished and upcoming jobs, schedule a new job or cancel a scheduled job. (administrator, wants to manage, jobs) (data manager, wants to manage, jobs) (administrator, to monitor, finished and upcoming jobs) (data manager, to monitor, finished and upcoming) (data manager, schedule, new job) (administrator, schedule, new job) (administrator, cancel, scheduled job) (data manager, cancel, scheduled job)</td>
</tr>
<tr>
<td>SiViOi</td>
<td>user and visitor should be able to conduct a search by providing either restaurant name, restaurant description. (user, to conduct search by providing, restaurant name) (user, to conduct search by providing, restaurant description) (visitor, to conduct search by providing, restaurant name) (visitor, to conduct search by providing, restaurant description)</td>
</tr>
</tbody>
</table>

\(^1\) We compared MRAalgo to ClausIE and OpenIE with the random sample of 50 sentences from software requirement specifications documents (SRS).
remained, in Princeton until his death), the former extraction is marked redundant. For MRAlgo, we took the confidence of the DP as obtained by the Spacy parser as the confidence of a triplet.

The importance of MRAlgo which distinguishes from other existing algorithms in retrieving relations from complex sentence with examples is evaluated in Table VI.

Our results are summarized in Tab VI which shows the total number of extractions for each method and dataset.

MRAlgo extracts all coherent triplets where ClauseIE could able to generate as well as extracting predicates where ClauseIE and OpenIE couldn’t able to extract. MRAlgo makes difference in extracting multiple triplets from sentence patterns with complex compound sentences with conjunctive nature as demonstrated in Evaluation table VI. ClauseIE prepositions for the selected sentence shown in table VI extracted only one predicate/verb, observed in C1, C2, C3, C4, C5 and C6. Similarly for OpenIE also extracted only one verb/predicate observed in O1 and O2.

MRAlgo triplets exploits every verb for all possible coherent patterns to uniquely identify every noun/noun phrase in the triplets. The predicate is constructed so as to give clear information regarding object of the triplet as well. This kind of approach increases the accuracy of Question and Answer system approaches to construct many questions from the corpus read. Objects in triplets B1, B2, B3, B4, B5 in MRAlgo is providing information about attributes of restaurant, which can uniquely identify restaurant. Object in triplets B6, B7, B8, B9, B10 is same and is explaining the governing functions of the object in the triplets, this information is useful in explaining what all ways a restaurant can be searched from the text search field.

The increase in recall is obtained because MRAlgo considers all verbs in a pattern, extracts non-verb-mediated propositions, detects non-consecutive constituents, processes coordinative, correlative and subordinative conjunctions, and outputs triplets with non-noun-phrase arguments.

We analyzed the effects of the sentence structures having Correlative conjunctions, Coordinating conjunctions and Subordinating conjunctions on the same collection of sentences. The performance of the algorithm was analyzed by randomly selecting 50 sentences from dataset whose embedded relations were manually extracted. Then we ran the algorithm on those sentences to extract triplets. Single relation extraction and multiple relation extraction algorithms extracted 140 and 151 triplets respectively, where as MRAlgo extracted 339 triplets from same number of sentences. Summarization metrics of triplets extracted for the selected random sample were shown in Table VII. While extracting triplets from sentences single and multi-relation extraction algorithms couldn’t able to retrieve even a single triplets from few sentences, whereas proposed algorithm extracted meaning full triplets even from those sentences. Example of such sentences were as follows Sentence 1: “sorting by restaurant name, specific dish or restaurant type the results should be ordered alphabetically”, single relation algorithm didn’t retrieved a single triplet. Proposed algorithm extracted three triplets and are

- (results’, ‘sorting by’, ‘restaurant name’)
- (results’, ‘sorting by’, ‘specific dish’)
- (results’, ‘sorting by’, ‘restaurant type’)

Sentence 2: “Since neither the mobile application nor the web portal have any designated hardware, these do not have any direct hardware interfaces”。 Multi-relation extraction algorithm couldn’t able to extract a single triplet from the above sentence. Proposed algorithm extracted following triplets

- (‘web portal’, ‘have’, ‘designated hardware’)
- (‘web portal’, ‘does not have’, ‘direct hardware interfaces’)
- (‘mobile application’, ‘does not have’, ‘direct hardware interfaces’)
- (‘mobile application’, ‘have’, ‘designated hardware’)

During the evaluation experiment, we identified that the false positives and false negatives were caused by some similar issues. Most false positives occurred when the sentences appeared in the description of the system to be developed. sorting by price the results should be ordered from cheapest to most expensive describes the objective of the system function rather than an actual relationship. The second triplet extracted by the algorithm is False positive. However, the proposed algorithm extracted all the valid relationships.

- (‘results’, ‘should be ordered from cheapest to’, ‘expensive’),
- (‘sorting by price’, ‘sorting by’, ‘price’),
- (‘sorting by price’, ‘should be ordered from cheapest to’, ‘expensive’)

VII. CONCLUSIONS AND FUTURE WORK

We propose MRAlgo which is Verb centric pattern based relation extraction algorithm. This approach explains an enhanced verb-based algorithm capable of extracting multiple relations embedded in a single sentence obtained from
### Table V. Sample Extracted Sentences for Evaluating Effectiveness

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A geographic information system (GIS), geographical information system, or geo-spatial information system is a system designed to capture, store, manipulate, analyze, manage and present all types of graphically referenced data</td>
</tr>
<tr>
<td>2</td>
<td>The GBIF Registry is an application, GBIF Registry manages the nodes, organizations, resources, and IPT installations registered with GBIF, making them discoverable and inter-operable</td>
</tr>
<tr>
<td>3</td>
<td>The mobile application will be used to find restaurants and view information about them while the web portal will be used for managing the information about the restaurants and the system as a whole</td>
</tr>
<tr>
<td>4</td>
<td>A link will direct the user to an information page, which includes a picture of the restaurant, the restaurant name, address, phone number, e-mail address, type of food, average price, restaurant description and a menu with name, description and price of the different dishes</td>
</tr>
<tr>
<td>5</td>
<td>Filtering options include increasing or decreasing the maximum distance, increasing or decreasing the maximum price, choosing a restaurant type, choosing a specific dish</td>
</tr>
<tr>
<td>6</td>
<td>A user should be able to conduct a search by providing either restaurant name, restaurant description, restaurant address, restaurant type or restaurant menu in the free-text search field</td>
</tr>
</tbody>
</table>

### Table VI. MRAalgo Extraction Evaluation

| Sentence 1: A user should be able to conduct a search by providing either restaurant name, restaurant description, restaurant address, restaurant type or restaurant menu in the free-text search field |
|---|---|
| **OpenIE Triplets:** |  |
| O1: (A user, should be, able to conduct a search by providing either restaurant name) |  |
| O2: (A user, should be, able to conduct a search) |  |
| **ClausIE Triplets:** |  |
| C1: (A user, should be, able to conduct a search by providing either restaurant name in the free-text search field) |  |
| C2: (A user, should be, able to conduct a search by providing restaurant description in the free-text search field) |  |
| C3: (A user, should be, able to conduct a search by providing restaurant address in the free-text search field) |  |
| C4: (A user, should be, able to conduct a search by providing restaurant type in the free-text search field) |  |
| C5: (A user, should be, able to conduct a search by providing restaurant menu in the free-text search field) |  |
| C6: (A user, should be, able to conduct a search) |  |
| **MRAalgo Triplets:** |  |
| B1: (user, to conduct search by providing, restaurant name) |  |
| B2: (user, to conduct search by providing, restaurant menu) |  |
| B3: (user, to conduct search by providing, restaurant address) |  |
| B4: (user, to conduct search by providing, restaurant description) |  |
| B5: (user, to conduct search by providing, restaurant type) |  |
| B6: (user, providing restaurant description in, text search field) |  |
| B7: (user, providing restaurant address in, text search field) |  |
| B8: (user, providing restaurant type in, text search field) |  |
| B9: (user, providing restaurant menu in, text search field) |  |
| B10: (user, providing restaurant name in, text search field) |  |
| B11: (user, providing, restaurant type) |  |
| B12: (user, providing, restaurant description) |  |
| B13: (user, providing, restaurant name) |  |
| B14: (user, providing, restaurant menu) |  |
| B15: (user, providing, restaurant address) |  |
| B16: (user, should be able to conduct, search) |  |
| B17: (user, should be, able to conduct) |  |

unstructured data. Given a requirement sentence written in natural language as input, the system processes it using dependency parsing which results in POS and dependency tokens. The proposed algorithm can handle complex sentences containing multiple relations such as conjunctive structure sentences (S1V1O1), (SnV1O1), (S1VnO1), (S1V1O)n, (SnV1O)n, (SnVnO)n, and (SiV6O)i. Each sentence was parsed and all verbs from the sentence were extracted and for every verb the subjects and its conjunctions were extracted and also all objects to the verb and its conjunctions were extracted. Every verb with their combination of subjects and objects were returned from algorithm as an output. Our multiple relation extraction algorithm achieved higher recall and higher precession in terms on total number of coherent extractions, when tested on the software requirement specification documents ALI [19] and NPT [20]. Although the comparison was performed over a relatively small sample, it shows a significant improvement of the
precision and recall rates of our algorithm over existing approaches.

<table>
<thead>
<tr>
<th>Table VII. MRAAlgo Evaluation Summarization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triples in OpenIE</td>
</tr>
<tr>
<td>Dataset</td>
</tr>
</tbody>
</table>

Our multiple relation extraction algorithm achieved higher recall and higher precession in terms on total number of coherent extractions, when tested on the software requirement specification documents ALI [19] and NPT [20]. Although the comparison was performed over a relatively small sample, it shows a significant improvement of the precision and recall rates of our algorithm over existing approaches.

REFERENCES


